Proceedings of the 1992 IEEE
International Conference on Robotics and Automation
Nice, France - May 1992

Posing Polygonal Objects in the Plane by Pushing

Srinivas Akella
Robotics Institute
Carnegie Mellon University

Matthew T. Mason
School of Computer Science
Carnegie Mellon University

Abstract

This paper studies the use of pushing actions with a fence to orient and translate objects in the plane. It describes a planner which is guaranteed to construct a sequence of pushing actions to move any polygonal object from any initial configuration to any final configuration. This planner, which utilizes an analysis of the mechanics of pushing an object, generates open-loop plans which do not require feedback sensing. These plans are guaranteed to succeed provided certain physical assumptions are met. We present results of experiments conducted to demonstrate the generated plans.

1 Introduction

The manipulation of objects restricted to motions in the plane is important in cases where the object cannot be grasped or it is more efficient to move the object in the plane. An example is planar parts transfer, where parts are to be moved from one position to another in the plane, often with a change in orientation.

In this paper, we develop a method to find open-loop plans, which do not require sensory feedback, to move a polygonal object from a known initial position and orientation to a goal position and orientation using linear pushing actions with a fence. We have implemented the method in a *Pose Planner*, and proven it complete; it is guaranteed to generate open-loop plans to move an arbitrary convex polygon from an arbitrary start position and orientation to an arbitrary goal position and orientation. We further describe experiments conducted to test the generated plans. Henceforth we refer to the combined position and orientation of the object as the *pose* of the object.

1.1 Related Work

Although the problem of simultaneously orienting and positioning an object has not been treated, there has been work on various aspects of orienting and positioning objects in the plane; see [Balorda, 1990, Mani and Wilson, 1985, Peshkin and Sanderson, 1988a]. [Mason, 1986] derived rules for the qualitative behavior of an object when subject to pushing actions. [Brost, 1988] utilized these results to develop an analysis of grasping actions with a parallel-jaw gripper. As an intermediate result, he analyzed the possible motions of an object being pushed by a fence to obtain the Push Stability Diagram. [Mani and Wilson, 1985] independently used Mason's rules to derive an Edge Stability Map, which was used to develop a planner to orient polygonal objects. [Peshkin and Sanderson, 1988b] contains bounds on the rate of rotation of an object being pushed, and the locus of the possible centers of rotation of the object. These are used to calculate the push distance guaranteed to orient an object. [Goldberg and Mason, 1990] simplified Brost's model for grasping by using only pushes normal to the face of the gripper.

2 The Problem

Given a polygonal object on a horizontal table at a known initial position and orientation (the start pose), we are to find a pushing plan to move it to a specified goal position and orientation (the goal pose). We call this the *Planar Pose Problem.* The pushing actions are executed by means of a fence, a flat edge used as a pusher. The pose is described by the orientation of the e1-edge, and position of the center of mass of the object. The start and goal poses may be arbitrary.

2.1 Outline of the Approach

To make the analysis tractable, we partially decouple the problem of achieving the desired orientation from that of achieving the desired position. We first consider the required change in orientation, and select a sequence of intermediate orientations, achieved by using *Reorientation pushes.* The intermediate orientations are selected based on an analysis of the mechanics of pushing an object with a fence. The reorientation pushes do not usually combine to move the object to the goal position. So we need a set of *Translation pushes* that can translate the object without changing its orientation. These can occur at the

0-8186-2720-4/92 $3.00 ©1992 IEEE

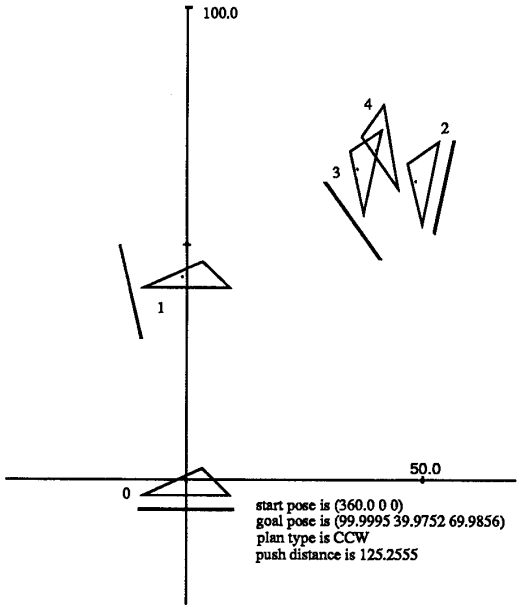

Figure 1: An example pose plan for a triangle. The zeroth and fourth poses are the start and goal poses respectively. The straight edges represent the fence.

initial, intermediate, and final orientations of the object. The lengths of the reorientation and translation pushes are determined so the net translation of the object is equal to the translation from the start position to the goal position. Thus the plans generated by the Pose Planner consist of a sequence of reorientation and translation pushes to be executed by the fence to achieve the required pose transformation. An example plan is depicted in Figure 1.

For our analysis to be valid, we assume:

1. Knowledge of the object's geometry and the location of its center of mass.
2. All objects are convex polygons. Non-convex polygons are equivalent to their convex hulls.
3. All motions are in the horizontal plane, which is assumed infinite in extent.
4. The fence is of sufficient length that the object does not roll off it.
5. All motions are quasi-static. That is, inertial forces are assumed negligible compared to frictional and applied forces.
6. Coulomb's law of friction describes all frictional interactions.
7. There is no slip between the object and the fence.
8. Support friction has a uniform friction coefficient.

We make no assumptions about the distribution of contact forces between the object and support plane.

3 The Mechanics of Pushing

3.1 Pushing Actions

Of the possible types of pushing actions, we restrict ourselves to *linear pushing actions*, where the fence starts moving from a point, with a specific orientation, in a constant direction for a specified distance. The result of a linear push depends on the angle between the fence and contact edge of the object, the distance the fence moves in contact with the object, and the direction of the push relative to the fence. Here we consider only pushes that are *normal*, or perpendicular, to the fence. Given the object's orientation, the orientation of the fence and the length of the push are then sufficient to describe a push.

3.2 The relation between pushing actions and polygon edges

When a fence lined up along a polygon edge executes a push, the polygon edge will either remain lined up with the fence or will rotate away. If the edge remains lined up with the fence, it is called a *Stable edge*. If it rotates away, it is an *Unstable edge*. To avoid uncertainty in the orientation of the object, the planner chooses pushes on only stable edges. Pushes with a fence along a stable edge of an object can be classified as follows (see Figure 2):

1. *Translational push*: The fence and object edge are always parallel and there is no change in the orientation of the object as the fence translates.

2. *Complete Reorientation push*: The push begins with an initial relative angle between the fence and object edge. By the end of the push, the object edge lines up with the fence. This results in a known change in object orientation. We refer to the minimum push distance guaranteed to align the object edge with the fence for a given relative angle as the *Peshkin distance* for that reorientation; it is obtained from [Peshkin and Sanderson, 1988b].

3. *Incomplete Reorientation push*: Here, unlike the complete reorientation push, the object edge does not line up with the fence by the end of the push. So the final orientation of the object is not known.

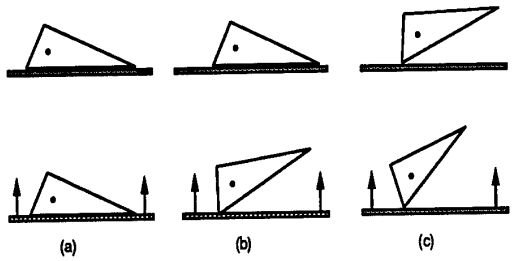

Figure 2: The different types of pushes (a) Translation push (b) Complete Reorient push (c) Incomplete Reorient push.

To avoid uncertainty in the object orientation, the planner uses only Complete reorientation pushes, hereafter referred to as reorient or reorientation pushes, and Translation pushes.

3.3 Object Classification

If the object is in the goal orientation, but not at the goal pose, it has to be translated to the goal position. So it is necessary that the goal orientation be attained in the *translation preimage* of the object at the goal pose, defined as the region from which the object can be translated to the goal pose without a change in orientation. The translation preimage, located at the center of mass, is obtained by finding the convex cone of the outward normals to the stable edges. It leads to the following classification of polygonal objects subject to pushing actions (see Figure 3):

1. **Uni-stables**: These are objects with only one stable edge, and hence their translation preimage is a ray extending to infinity in one direction from their center of mass.

2. **Biplus-stables**: These objects have more than one stable edge such that the normals to the stable edges do not span $\Re^2$, the plane. The translation preimage of a biplus-stable is a cone centered at its center of mass.

3. **Nice-stables**: These objects have at least three stable edges such that the normals to the stable edges span $\Re^2$. Therefore the translation preimage of these objects is the entire plane.

This analysis implies that uni-stables and biplus-stables cannot always be translated between any two positions without a change in orientation. (See Figure 4 for an example.)

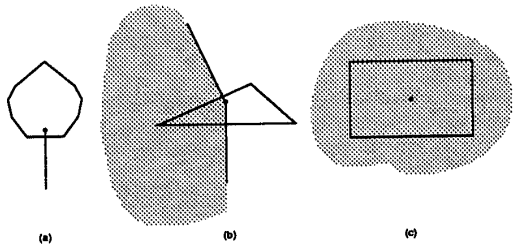

Figure 3: Examples of the three object types with their translation preimages shown shaded. (a) Uni-stable (b) Biplus-stable (c) Nice-stable.

3.4 Stable edges and the Angle-eating heuristic

We can derive the *Push Stability Diagram* (PSD) for a convex polygonal object [Brost, 1988]. The PSD indicates the stable edges of the polygon, and gives, for a particular rotation direction, the maximum angle between a stable edge and the fence for which the edge can be stably reoriented onto the fence. Since we permit only pushes involving stable edges, the *maximum reorient angle* (defined in Figure 6) is the maximum allowable angle between the fence and edge. Thus, for each stable edge, we find the maximum reorient angles permitted for Clockwise (CW) and Counterclockwise (CCW) rotations. The maximum reorient angle is always less than 90 degrees.

Using the maximum reorient angle can result in large Peshkin distances and sometimes, failure of the reorient push; a reorient push is a failure when it does not result in the stable edge lining up with the fence. To avoid this, we use the maximum reorient angle less a *margin angle*, which provides a safety margin. This is the magnitude of the reorientation achieved by each reorient push, and is called the *step-angle*. The step-angle is also always less than 90 degrees.

The *Angle-eating heuristic*, used to generate reorient sequences, seeks to minimize the number of pushes. The reorientation strategy requires that the object go through a sequence of intermediate orientations to at-

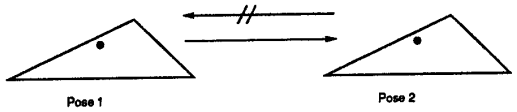

Figure 4: The triangle can be translated from pose 1 to pose 2 since pose 1 lies in its translation preimage at pose 2. The converse is not true.

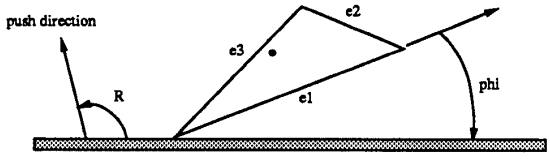

Figure 5: The notation used in the Push Stability Diagram.

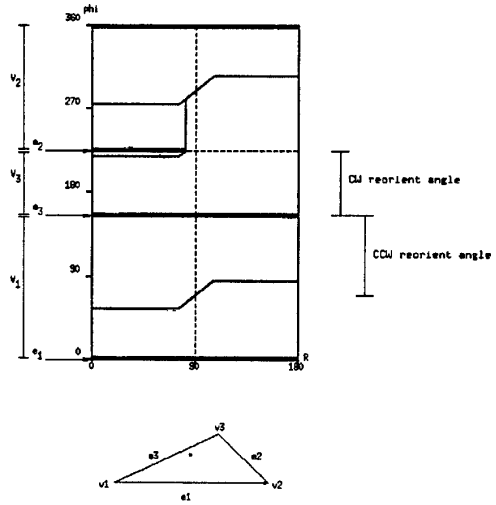

Figure 6: The Push Stability Diagram for the triangle. The horizontal axis and vertical axis represent the pushing direction and polygon e1-edge orientation relative to the fence pushing surface. For normal pushes, the points of interest are on the dotted vertical line at 90 degrees. The figure wraps around vertically at 0/360 degrees. Convergent lines, which correspond to stable edges, are indicated by double bold lines. Divergent lines are represented by single bold lines. Since this PSD has only two distinct convergent lines, the triangle is a Biplus-stable. CW rotations correspond to moving downwards along the diagram, and CCW rotations correspond to moving upwards. The *maximum CW reorient angle* for a given stable edge is found by moving downwards along the PSD to the edge from the nearest divergent line or (stable or unstable) edge. The *maximum CCW reorient angle* is found similarly by moving upwards. The CW and CCW reorient angles for edge e3 are indicated. From [Brost, 1988].

tain the goal orientation. These intermediate orientations depend on the magnitude of the step-angle. The heuristic assumes that the larger the step-angle, the fewer the pushes required to get the object to the goal orientation, and hence to the goal pose. To obtain the largest step-angle, we perform reorient pushes on the stable edge which permits the largest maximum reorient angle for the chosen direction sense. Then the orientation change is divided into an integer number of step-angle reorientations, and if necessary, a final smaller reorientation.

4 Determining the pushes

A pushing plan to pose an object consists of a sequence of Reorient and Translation pushes. The Angle-eating heuristic provides a sequence of fence orientation angles, which define a corresponding sequence of reorient push directions. For the the reorientations to be complete, the pushes need to have a minimum magnitude (the Peshkin distance). These constraints coupled with the push directions partially determine the reorient pushes. At this stage, we need to select translation pushes and determine the magnitudes of the reorient pushes to ensure that the object reaches the goal position in the desired goal orientation.

4.1 The Pose vector equation

To determine the magnitudes of the pushes, we treat them as vectors in the plane. At a given orientation of the object, translation pushes can occur only in the directions of the unit normals into the stable edges. Let this set of unit normal push vectors be denoted by $\{\hat{n}_{ij}\}$, where the subscripts refer to the ith orientation and jth stable edge. Let a_{ij} represent the length of the push along the normal $\hat{n}_{ij}$; magnitude a_{ij} is non-negative. Let the set of unit reorient push vectors be denoted by $\{\hat{p}_i\}$, and let b_i be the length of the push along $\hat{p}_i$. The minimum magnitude of b_i is the Peshkin distance for the ith reorientation.

The *push vector diagram* is a graphical representation of the unit push vectors. It is defined, for a given reorient sequence, as the union of the $\hat{p}_i$ vectors of the chosen stable edge and the $\hat{n}_{ij}$ vectors of all stable edges at all orientations in the sequence. See Figure 8 for examples.

We have assumed that the object does not slide relative to the fence. So during a reorient push, the object rotates about a fixed point on the fence as the fence translates. As the object rotates, its center of mass moves relative to the fence. The *Center of Mass vector* $\vec{c}_i$ describes this relative motion, and for a given

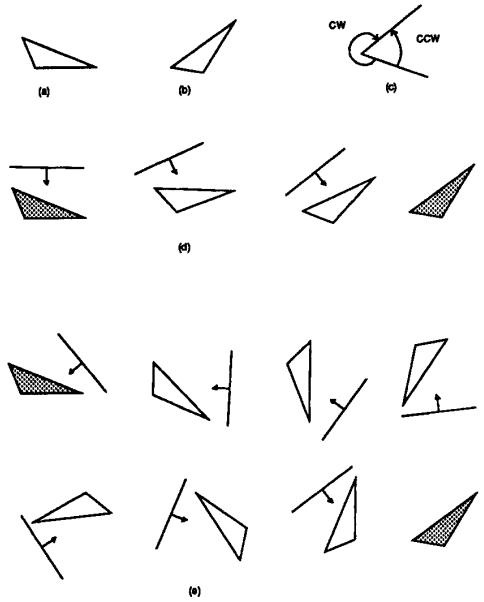

Figure 7: Illustrating two sequences to reorient an object. The arrows indicate the reorient push directions. (a) Start orientation (b) Goal orientation (c) Net change in orientation for CCW and CW rotations (d) CCW Reorient push sequence (e) CW Reorient push sequence.

initial and final orientation is determined by the geometry.

The magnitudes of the push vectors, $\{a_{ij}\}$ and $\{b_i\}$, are to be determined so that all the vectors combine to obtain the *Translation vector* $\vec{T}$, which is the vector from the start center of mass position to the goal center of mass position. The equation relating all the relevant vectors is the *Pose vector equation*

$$\sum_{i=0}^{r} \sum_{j \in S} a_{ij} \hat{n}_{ij} + \sum_{i=1}^{r} b_i \hat{p}_i + \sum_{i=1}^{r} \vec{c}_i = \vec{T}$$

where r is the number of reorientations required to reach the goal orientation and S is the set of stable edges of the object.

A solution to this equation is to be found which satisfies the *Non-negativity constraints*

$a_{ij} \geq 0, \ \forall i = 0, \ldots, r, \ \forall j \in S$

and the *Peshkin constraints*

$b_i \geq m_i, \forall i = 1, \ldots, r$

where m_i is the Peshkin distance for the *ith* reorientation.

The next section describes a method to determine

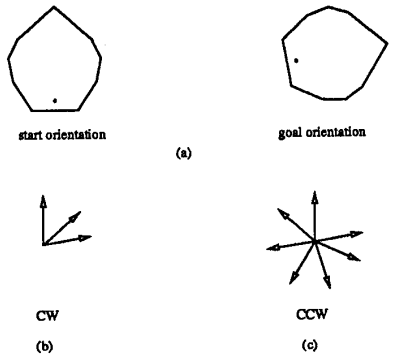

Figure 8: (a) The start and goal orientations. (b) CW push vector diagram. (c) CCW push vector diagram.

the magnitude terms a_{ij} and b_i of the corresponding vectors $\hat{n}_{ij}$ and $\hat{p}_i$.

4.2 A Linear Programming solution

The magnitudes of the push vectors can be determined in a simple and elegant manner by developing a Linear Programming (LP) formulation [Chvatal, 1983]. Once a reorientation sequence is chosen, the $\vec{c}_i$ vectors are determined, and the Pose vector equation can be written as the *Modified Pose vector equation*

$\sum_{i=0}^{r} \sum_{j \in S} a_{ij} \hat{n}_{ij} + \sum_{i=1}^{r} b_i \hat{p}_i = \vec{T}'$

where $\vec{T}' = \vec{T} - \sum_{i=1}^{r} \vec{c}_i$.

We scalar-multiply this vector equation by the unit vectors along and perpendicular to $\vec{T}'$, $\hat{t}'$ and $\hat{t}'_{perp}$ respectively, to obtain two scalar constraints. In addition, we have a set of magnitude constraints linear in a_{ij} and b_i.

The particular push sequence obtained depends on the optimizing criterion used. We minimize the total push distance, which is the sum of the a_{ij} and b_i terms. So the LP formulation of the problem is:

Minimize $\sum_{i=0}^{r} \sum_{j \in S} a_{ij} + \sum_{i=1}^{r} b_i$

such that

$\sum_{i=0}^{r} \sum_{j \in S} a_{ij} (\hat{n}_{ij} \cdot \hat{t}') + \sum_{i=1}^{r} b_i (\hat{p}_i \cdot \hat{t}') = T'$

$\sum_{i=0}^{r} \sum_{j \in S} a_{ij} (\hat{n}_{ij} \cdot \hat{t}'_{perp}) + \sum_{i=1}^{r} b_i (\hat{p}_i \cdot \hat{t}'_{perp}) = 0$

$a_{ij} \geq 0, \ \forall i = 0, \ldots, r, \ \forall j \in S$

$b_i \geq m_i, \forall i = 1, \ldots, r$

A solution to the above LP problem minimizes the push distance for the chosen set of intermediate orientations and provides a feasible pose plan.

5 The Pose Planner

The Pose Planner looks for plans that involve only translation pushes, CW reorient pushes and translation pushes, and CCW reorient pushes and translation pushes. These are the *Translation, CW,* and *CCW pose plans* respectively. The planner attempts to generate CW and CCW pose plans for each stable edge, and a Translation pose plan when the start and goal orientations are equal. The "best" plan (having the lowest value of the objective function) among all the plans is chosen the pose plan. Infeasible solutions are indicated by extremely large values of the objective function.

Sometimes the above procedure does not yield a feasible solution for Uni-stables. We refer to such a case as a *Singularity.* When this occurs, the planner uses the *CW* and *CCW Singularity handlers* to generate modified CW and CCW reorient sequences. A modified reorient sequence consists of the usual reorient sequence preceded by a reorientation in the opposite direction. So the *CW singularity plan* consists of a CCW reorient push preceding a sequence of CW reorient pushes; similarly for the *CCW singularity plan.* This is guaranteed to result in a solution for each main rotation direction, of which the better one is selected.

The planner can be recast to minimize the number of pushes using a Mixed Integer Linear Programming (MILP) formulation [Nemhauser, 1988]. The current planner can be extended to find plans within a finite rectangular surface. However, it is then not guaranteed to be complete. The planner can also be modified to use different stable edges to perform reorient pushes on, or to use combinations of CW, CCW, and Translation pushes, but this would increase the search space.

5.1 A Completeness Criterion

We wish to prove that the Pose Planner is complete, that is, that it can find a pose plan for an arbitrary pose problem for an arbitrary polygon. A proof of completeness is equivalent to a proof that for each polygon type, at least one sequence of translation and reorient pushes generated by the planner will satisfy the Pose vector equation and the Non-negativity and Peshkin constraints. Since pushes are uni-directional, we can consider only positive linear combinations of the unit push vectors, and need to show that at least one set of unit push vectors positively spans the plane. In [Davis, 1954] is a theorem we refer to as the *Positive Spanning theorem*: A set of vectors $\{\vec{a}_1, ..., \vec{a}_r\}$ positively spans $\Re^n$ if and only if, for every non-zero vector $\vec{b}$, there exists an $i \in 1, \ldots, r$, such that $\vec{b} \cdot \vec{a}_i > 0$.

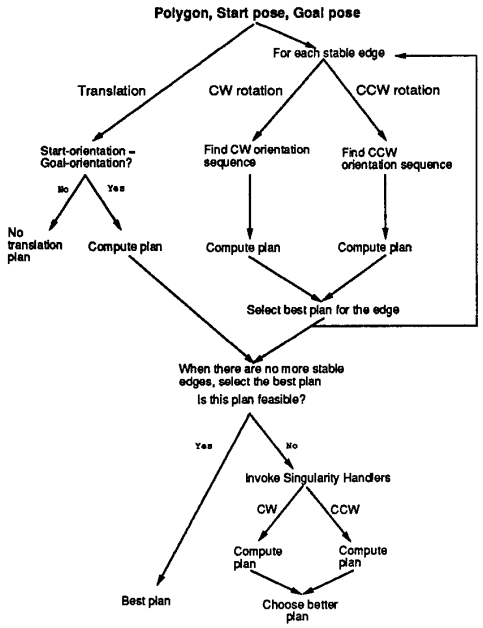

Figure 9: An algorithmic description of the Pose Planner. Given information on the polygon, and the start and goal poses, it outputs the best plan, as defined by the objective function.

If the unit push vectors are so arranged that at least one of them forms an acute angle with any arbitrary vector in the plane, they will positively span $\Re^2$. Since the step-angle is always less than 90 degrees, a reorientation angle greater than 180 degrees guarantees that the unit push vectors chosen by the planner satisfy the conditions of the above theorem. See Appendix for details.

6 Implementation and Experiments

The Pose Planner was implemented in Common Lisp. The inputs to the planner are the start and goal poses, and a geometric description of the object, in terms of its vertices and center of mass. It utilizes the code to generate the push stability information described in [Brost, 1988] and a linear programming package, LINDO [Schrage, 1981]. The planner takes 30 to 90 seconds on an IBM RT to find the best plan.

When generating and executing plans, we can vary the margin angle, and the *push offset,* which is the distance the fence is offset from the polygon when starting a push. These are useful in handling limited un-

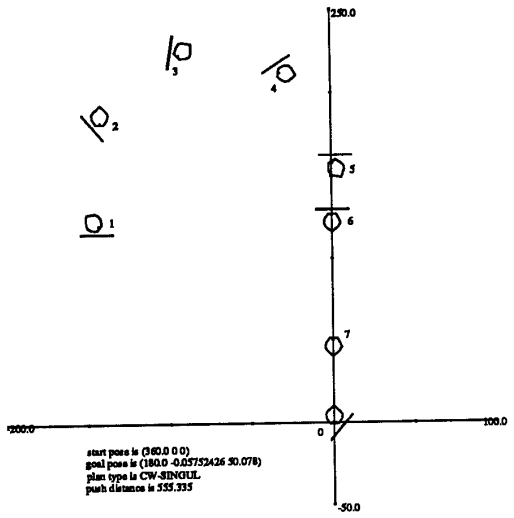

Figure 10: A plan generated to handle the Uni-stable singularity case.

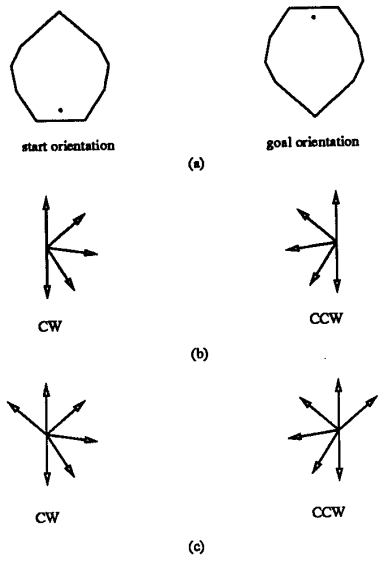

Figure 11: A singularity case. (a) Start and goal orientations. (b) These CW and CCW push vector diagrams do not positively span the plane. (c) The CW and CCW push vector diagrams generated by the singularity handler positively span the plane.

certainty in the orientation and position of the object. We currently use a margin-angle of 7 degrees and a push offset of 3 cm. Since the calculated Peshkin distances were observed insufficient to achieve reorientation, they were multiplied by a safety-factor of 1.15.

With the above values of the margin-angle and push offset, six plans were executed on a Puma 560 robot for a total of over 80 trials. The Puma, with a fence attached to its end-effector, executes pushing actions on a horizontal table. The fence is a piece of delrin coated with high-friction sandpaper. Objects belonging to the three types were constructed of delrin. They were made fairly small, about 1.5 to 3 cm in diameter, for the plans to be inside the Puma workspace. While the object usually attained the goal orientation, position errors of 2 to 3 mm over a translation distance of 15 to 20 cm were commonly observed. The observed errors were a result of the motion of the object along the fence due to robot vibration, and the positioning inaccuracy of the Puma.

7 Conclusion

In this paper we have introduced a useful classification of polygonal objects subject to linear pushing actions, and shown that the Planar Pose problem can be solved under certain assumptions. Further, we have implemented a planner for this problem and proved its completeness. Sample plans executed by a robot demonstrate the feasibility of this method. For a detailed discussion, see [Akella and Mason, 1992].

An important aspect of our approach is the use of a compact set of actions for the task. An alternative approach is to use a more extensive set of actions, such as non-normal linear pushes, or the rotational pushes in [Lynch, 1992].

Plans using the actions described here are sensitive to uncertainty. To generate robust plans, it is important to model uncertainty in the initial pose of the object, in the pushing actions of the robot, and in the motion of the object along the fence. Determining if "nearby" start poses behave similarly to a given plan could be useful in obtaining bounds on the permissible pose uncertainty for the plan. An interesting issue for future work is the coupling of empirical learning techniques with the analytical tools developed here to generate more robust plans.

Acknowledgments

Thanks to Randy Brost for generously sharing code and insights. Discussions with Alan Christiansen, Mike Erdmann, N.R. Natraj, Fred Solomon, and Manipulation lab members helped shape the work. This work was supported by the Robotics Institute, Carnegie Mellon University.

References

[Akella and Mason, 1992] S. Akella and M. T. Mason. Posing Polygonal Objects in the Plane by Pushing. Forthcoming Technical Report, Robotics Institute, Carnegie Mellon University, 1992.

[Balorda, 1990] Z. Balorda. Reducing uncertainty of objects by robot pushing. In *Proceedings, IEEE International Conference on Robotics and Automation*, May 1990.

[Brost, 1988] R. C. Brost. Automatic grasp planning in the presence of uncertainty. *International Journal of Robotics Research*, Vol.7, No.1, February 1988.

[Chvatal, 1983] V. Chvatal. *Linear Programming.* W.H.Freeman and Co., 1983.

[Davis, 1954] C. Davis. Theory of Positive Linear Dependence. *American Journal of Mathematics*, Vol. 76, 1954.

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

[Lynch, 1992] K. M. Lynch. The mechanics of fine manipulation by pushing. In *Proceedings, IEEE International Conference on Robotics and Automation*, May 1992.

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

[Mason, 1986] M. T. Mason. Mechanics and Planning of Manipulator Pushing Operations. *International Journal of Robotics Research*, Vol. 5, No. 3, Fall 1986.

[Nemhauser, 1988] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley & Sons, 1988.

[Peshkin and Sanderson, 1988a] M. A. Peshkin and A. C. Sanderson. Planning robotic manipulation strategies for workpieces that slide. *IEEE Journal of Robotics and Automation*, Vol.4, No.5, October 1988.

[Peshkin and Sanderson, 1988b] M. A. Peshkin and A. C. Sanderson. The motion of a pushed, sliding workpiece. *IEEE Journal of Robotics and Automation*, Vol.4, No.6, December 1988.

[Schrage, 1981] L. E. Schrage. *User's manual for LINDO.* The Scientific Press, 1981.

Appendix: Proof of completeness

Uni-stables: Since $\hat{n}_{is} = \hat{p}_i$ (subscript s refers to the stable edge), the Pose vector equation can be replaced by

$a_{0s}\hat{n}_{0s} + \sum_{i=1}^{r}(a_{is} + b_i)\hat{p}_i + \sum_{i=1}^{r}\vec{c}_i = \vec{T}$.

The Peshkin distance for the ith reorientation is m_i. So $m_i > 0$. Let $a_{is} + b_i = m_i + e_i$, where $e_i \geq 0$, and is the extra distance along $\hat{p}_i$.

So, the above vector equation is equivalent to

$a_{0s}\hat{n}_{0s} + \sum_{i=1}^{r} e_i\hat{p}_i = \vec{T''}$,

where $\vec{T''} = \vec{T} - \sum_{i=1}^{r}\vec{c}_i - \sum_{i=1}^{r} m_i\hat{p}_i$.

If the planner always generates at least one reorientation sequence such that $\hat{n}_{0s}$ and $\{\hat{p}_i\}$ positively span the plane, this vector equation will always be satisfied. Two cases can occur:

1. *The reorientation is not 180 degrees* : Since the reorientation in one of the rotation directions is greater than 180 degrees, the Positive Spanning theorem is satisfied. In Figure 8 for example, the unit vectors associated with CCW rotation positively span the plane.

2. *The reorientation is 180 degrees or nearby*: Since neither the CW nor CCW unit push vectors positively span the plane for this case (see Figure 11 (b)), we cannot guarantee existence of a feasible pose plan. To obtain a set of unit reorient push vectors that positively span the plane, we first make an additional reorient push in the rotation direction opposite to the remaining reorient pushes. The amount of rotation in the opposite direction is chosen equal to the minimum of the CW and CCW step-angles for the chosen pushing edge. This modified procedure always results in a solution for each of the main rotation directions (see Figure 11 (c)).

Biplus-stables: In the singularity case, since the normal vectors of a biplus-stable form a cone, one of the unit push vectors sets generated by the CW and CCW routines satisfies the Positive Spanning theorem.

Nice-stables: Since the normals to the edges span $\Re^2$, a combination of translation vectors can always be found to move the object in the goal orientation to any position in the plane.