Shape Tolerance in Feeding and Fixturing

Jingliang Chen*, University of California at Berkeley, USA
Ken Goldberg, University of California at Berkeley, USA
Mark H. Overmars, University of Utrecht, The Netherlands
Dan Halperin, Tel Aviv University, Isreal.
Karl F Böhringer, University of California at Berkeley, USA
Yan Zhuang, University of California at Berkeley, USA

Abstract

Parts are not ideal. Designers and machinists cope with variation in part shape by specifying tolerance zones around a nominal part geometry: all parts that fit within the zone form a tolerance class. In this paper we consider the issue of shape tolerance in two contexts. First, we consider the problem of *feeding* (orienting) convex polygonal parts on a conveyor belt with a part-specific sequence of fence angles [1]. Second, we consider the problem of *fixturing* convex polygonal parts using a right angle fixture and one clamp [2]. The challenge is to define appropriate tolerance classes and to argue that a solution – a feeding strategy or a fixture – is guaranteed for all parts in the tolerance class.

We propose two new parametric tolerance classes. For each, we give an $O(n)$ time algorithm for testing if an n-sided part is in the class. For *feeding* we give an $O(n^2)$ time algorithm to compute the maximum radius of a circular tolerance zone around each vertex. Numerical experiments provide evidence that this bound is tight. For *fixturing* we give an $O(1)$ time algorithm to compute the maximum dimensions of rectangular tolerance zones. We implemented both algorithms and report experimental results.

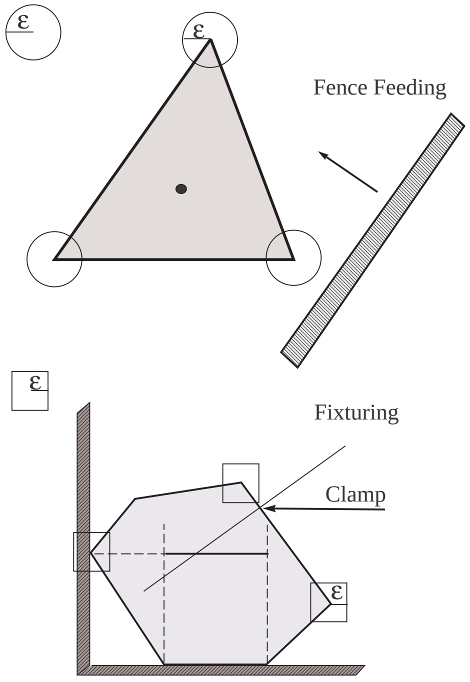

Figure 1: *Tolerance classes for feeding and fixturing.*

1 Introduction

Geometric tolerance is increasingly important as parts and products shrink, yielding relatively larger variations in part shape. Such variations are inevitable due to physical fabrication; all parts vary at some microscopic level. New processes such as MEMS (Micro Electro Mechanical Systems) and stereo lithography impose relatively large variations in part geometry. Characterizing allowable shape tolerance is important for product functionality; it is less widely recognized that variations in part shape can also cause manufacturing processes to fail. In this paper we consider two specific manufacturing processes, fence feeding and modular fixturing, and develop new algorithms for characterizing shape tolerance in these contexts.

We build on the geometric algorithms for feeding and fixturing that assume exact part geometry as input.

* jlchen@ieor.berkeley.edu

In P. Agarwal, L. Kavraki, M. Mason, editors, Robotics: The Algorithmic Perspective, A. K. Peters, Ltd. 1998.

For each process, we define a tolerance class $\Delta(P, \varepsilon)$ based on a nominal part P and a variation parameter ε as shown in Figure 1. Both classes are defined so that we can check membership in linear time. Given a nominal part P, we compute a solution, a feeding strategy or fixture. We then compute the largest allowable ε such that the solution will work for all parts in $\Delta(P, \varepsilon)$.

2 Related Work

A part feeder is a device that accepts parts in random orientation and outputs them in a unique orientation. Currently, the design of parts feeders is a black art that is responsible for up to 30% of the cost and 50% of workcell failures [3, 4]. An algorithm for feeding polygonal parts was proposed in [5] and subsequently refined in a series of papers [6, 7, 1]. The model we consider here orients parts of a given shape as they are pushed by a sequence of fences. Thus a solution is a sequence of fence angles.

A fixture is a device that locates and holds parts during assembly, inspection, or machining. Modular fixturing systems typically include a lattice of holes with precise spacing and an assortment of precision locating and clamping modules that can be rigidly attached to the lattice. The model we consider here clamps parts with a right angle fixture and one translating clamp [2]. For fixturing, a solution is an orientation for the part and a position for the clamp.

For reviews of semantic issues in tolerancing, see [8, 9, 10]. Parametric tolerance models allow certain parameters to vary within a given range. On the other hand, geometric tolerance zones require only that the part fit within a defined region or zone [8]. Actual part geometry can be quite gnarly while remaining within the zone. It is extremely difficult to characterize part behavior under these conditions although bounds on behavior of kinematic pairs has been the subject of a study by Joskowicz, Sacks and Srinivasan [11].

Yap and Chang [12] consider geometric metrology: given a set of sample points on the boundary of a closed curve, decide if the curve lies within a given geometric tolerance zone. Even for a (one-dimensional) line segment, conservative classification policies may not exist; the authors illustrate that deterministic policies and statistical decision rules can be non-trivial even in one dimension. The parametric tolerance models defined below permit exact classification in linear time.

Latombe, Wilson and Cazals consider tolerancing in the context of assembly [13]. They propose a parametric tolerance model similar to ours in that they assume part edges are straight. To make the analysis manageable, they add the requirement that edges/faces maintain their relative orientation: i.e., edges in the tolerance class always remain parallel to the nominal edge. For a given set of tolerance specifications, the authors give polynomial time algorithms to decide if an assembly sequence exists for an assembly of polygonal or polyhedral parts. For our applications, we explicitly compute the radius of an allowable tolerance zone around each vertex.

Perhaps the work closest to ours is Akella and Mason [14, 15]. They consider fence and push-squeeze plans for orienting polygonal parts based on a similar framework. One difference is that their parametric tolerance class defines a circular tolerance zone around the center of mass. They note that determining a bound on the possible center of mass of a uniform mass polygon, in terms of bounds on its vertices, is an open problem. We finesse this problem by defining a tolerance class whereby vertices are defined relative to a coordinate frame at the center of mass.

Given the radius of a circular tolerance zone around each vertex and the center of mass, Akella and Mason compute action ranges and uses breadth-first search of an AND/OR tree to check if a solution exists. We extend their framework for feeding with a new model for shape tolerance; rather than testing if a given radius will succeed, we estimate the maximum radius of the tolerance zones.

Donald [16] considered parametric shape variation in the context of motion planning by adding dimensions to configuration space. Our work is a step toward bounding variation in these dimensions. Brost and Peters [17] propose a different model of shape tolerance for modular fixturing where contact normals are allowed to vary within a cone of given half angle. Kavraki considered variation in part shape in the context of orienting parts with an elliptical planar force field. She was able to bound the final orientation of a part in terms of the shape difference between parts P and P': $P - P' \cup P' - P$ [18].

In this paper, we define parametric tolerance classes for feeding and fixturing. We give algorithms for finding bounds on the size of the tolerance zone and algorithms for checking if a part is in the class.

3 Shape Tolerance for Feeding

Polygonal parts can be fed (oriented) as they come into contact with fences on a conveyor belt [19, 20, 21, 1]. Let P be the nominal convex polygonal part with n vertices and center of mass c. We assign a fixed coordinate frame to P, with c as the origin of the coordinate frame and the x-axis aligned with the desired nominal final orientation of the part after feeding.

All directions are expressed relative to this coordinate frame as shown in Figure 2. Vertices are counterclockwise labeled $v_1 \dots v_n$ and e_i is the edge connecting v_i and v_{i+1}; the edge e_n connects v_n and v_1. Let α_{v_i} be the direction of the ray cv_i in the coordinate system and α_{e_i} be the direction of the normal from c to the line containing the edge e_i. Let $\alpha_{r_i} = \alpha_{e_i} - \alpha_{v_i}$, $\alpha_{l_i} = \alpha_{v_{i+1}} - \alpha_{e_i}$; these angles play a crucial role in our analysis.

3.1 A Tolerance Class for Feeding

For feeding, we define a parametric tolerance class $\Delta(P, \varepsilon)$ based on a nominal part P with center of mass c, as shown in Figure 3. We assume that part edges are always straight. Each vertex of P has a circular tolerance zone of radius ε. A part P' with center of mass c' is in $\Delta(P, \varepsilon)$ if we can position and orient P' such that c' is aligned with c and all of P''s vertices are inside the tolerance zones around the corresponding vertices of P. (Note that we assume a given assignment of vertices.) We assume that topologies of the convex hull of all the parts in $\Delta(P, \varepsilon)$ are the same. We can test whether or not a polygon P' is in $\Delta(P, \varepsilon)$ by computing an angular interval $\Omega_i^{'}$ for each vertex $v_i^{'}$ such that rotating $P^{'}$ by $\omega \in \Omega_i^{'}$ will move $v_i^{'}$ into the ε disk around v_i, or will keep it inside the disk if it is already inside the disk. If $\cup_i \Omega_i^{'} \neq \emptyset$, then $P^{'} \in \Delta(P, \varepsilon)$. The complexity of this test is $O(n)$.

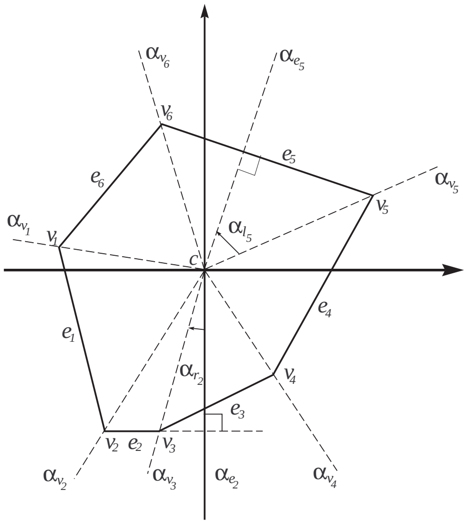

Figure 2: *Notation for feeding*

3.2 The Push Function and Push Plans

As shown in 4, the *push function* characterizes the mechanics of a part pushed by a frictionless planar fence [22, 5]. We adopt the notation of [1]. In most cases, parts will start to rotate when pushed. We define the push direction as the direction of the normal from c to the pushing fence. If pushing in a certain direction does *not* cause the part to rotate, we refer to the corresponding push direction of as an *equilibrium* push direction. There are two types of equilibrium push direction, stable equilibrium push direction and unstable equilibrium push direction. If pushing does change the orientation, then this rotation changes the orientation of the pushing device relative to the part. We assume that pushing continues until the part stops rotating and settles in a stable equilibrium pose.

Edge e_i is called a *stable edge* if α_{e_i} corresponds to a stable equilibrium push direction, and e_i is called unstable otherwise. Similarly vertex v_j is called an

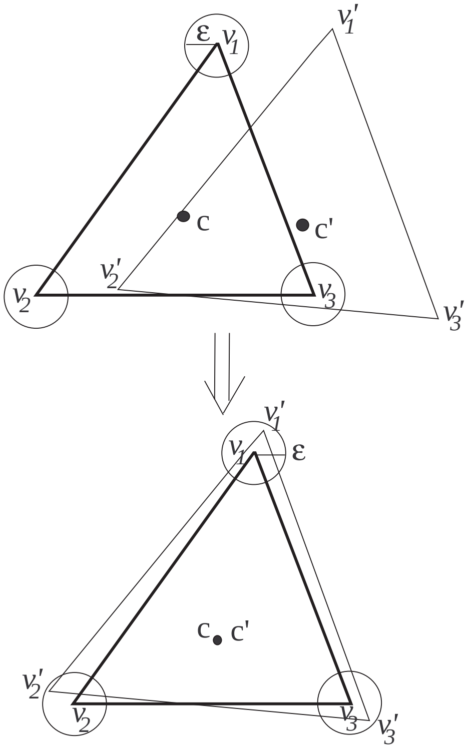

Figure 3: *Check if candidate part $P' \in \Delta(P, \varepsilon)$*

.

equilibrium vertex if α_{v_j} corresponds to an equilibrium push direction. In the case of a convex polygon, all stable push directions correspond to stable edges and all the equilibrium push directions corresponding to vertices are unstable. Edge e_i is stable if and only if $\alpha_{r_i} > 0$ and $\alpha_{l_i} > 0$.

The push function $p : [0, 2\pi) \rightarrow [0, 2\pi)$ links every orientation ϕ to the orientation $p(\phi)$ in which the part P settles after being pushed by a jaw with initial push direction ϕ. During the pushing, the rotation of the part causes the push direction of the jaw to change. The final orientation $p(\phi)$ of the part is the push direction of the jaw after the part has settled. The equilibrium push directions are the fixed points of p and, as we stated above, they all correspond to stable edges.

The push function p of a polygonal part consists of steps, which are intervals $I \subset [0, 2\pi)$ such that for all $\phi \in I$, $p(\phi) = C$ for some constant $C \in I$; see Figure 4. The steps of the push function are easily constructed from the α_{l_i}'s and α_{r_i}'s. The interval I is bounded by two consecutive unstable equilibrium push directions. All directions strictly inside the interval map onto the stable equilibrium push directions corresponding to a stable edge. (Note that the direction C itself maps onto C because it is the direction of the stable edge.) As a result, the number of steps in the push function equals the number of stable edges.

In preparation for the next section, we define for each stable edge e_i two open intervals $l(e_i) = \{\phi < \alpha_{e_i} | p(\phi) = \alpha_{e_i}\}$ and $r(e_i) = \{\phi > \alpha_{e_i} | p(\phi) = \alpha_{e_i}\}$. We refer to these intervals as e_i's left and right environment respectively. The interval $l(e_i)$ corresponds to the half-step left of α_{e_i} and $r(e_i)$ corresponds to the half-step right of α_{e_i} (Figure 4). The open intervals $l(e_i)$ and $r(e_i)$ are maximal intervals without equilibrium orientation.

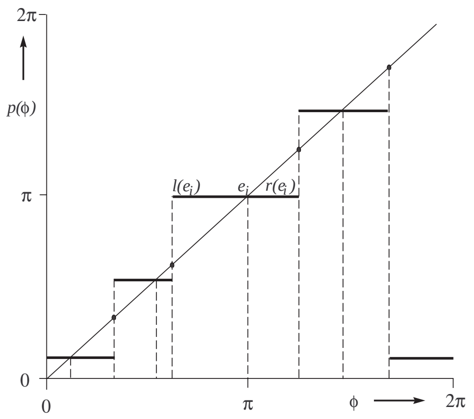

Figure 4: *The push function for a polygonal part. Each step corresponds to a stable edge and is bounded by two consecutive unstable equilibrium push directions which correspond to the directions of equilibrium vertices.*

We use the abbreviation p_α to denote the (shifted) push function defined by

$$p_\alpha(\phi) = p((\phi + \alpha) \bmod 2\pi)$$

for all $\phi \in [0, 2\pi)$. Note that $p_\alpha(\phi)$ is the final orientation of a part in initial orientation ϕ after a reorientation by α followed by a push. We can now define a *push plan*.

Definition 3.1 *A push plan is a sequence* $\alpha_1, \ldots, \alpha_m$ *such that* $p_{\alpha_m} \circ \ldots \circ p_{\alpha_1}(\phi) = \Phi$ *for all* $\phi \in [0, 2\pi)$ *and some fixed* $\Phi \in [0, 2\pi)$.

3.3 Push Planning Algorithms

Goldberg [5] showed that a push plan exists for any polygonal part and gave an $O(n^2)$ algorithm for finding the shortest such plan. Chen and Ierardi [7] gave a different algorithm based on finding a basic pushing action that repeatedly applies one critical pushing angle to orient the part. The critical pushing angle depends on the length of the two largest open left half intervals in the push function. The same is true for right intervals but, without loss of generality, we only consider left push plans here. Let Ψ be a range of angles defined by the difference in length of these half intervals. Any angle in Ψ will serve as the critical pushing angle. This range can compensate for variation in part shape.

Consider the open intervals $l(e_i)$ defined in the previous section. Let α be the length of the longest interval in the set. We assume, without loss of generality, that e_1 in P is a stable edge with $|l(e_1)| = \alpha$, and we only consider the case in which P has no second stable edge e_i $(i \neq 1)$ satisfying $|l(e_i)| = \alpha$. Each basic action consists of a reorientation of the jaw by an angle of $-(\alpha - \mu)$, with $\mu > 0$ such that $\alpha - \mu > |l(e_i)|$ for any stable edge $e_i \neq e_1$, and a subsequent application of the jaw. (The same arguments will apply for $r(e_i)$ with reorientation by an angle of $\alpha - \mu$ instead of $-(\alpha - \mu)$.)

Note that a reorientation of the jaw by $-(\alpha - \mu)$ corresponds to a change of the push direction by $\alpha - \mu$. Every basic action puts the part into a stable equilibrium orientation. If P is a polygonal part, then the stable equilibrium orientations occur at isolated points in $[0, 2\pi)$. After each basic action, the number of possible part orientation is finite. Let us label the m stable edges $e_{s_1}, \ldots, e_{s_m}$ in order of *increasing* edge index with $s_1 = 1$. After the first application of the jaw, the part P can be in the orientation of any stable edge $e_{s_1}, \ldots, e_{s_m}$. Chen and Ierardi show that every next basic action eliminates the last stable edge in the sequence. So an $m + 1$ step push plan $p_{\alpha-\mu} \circ \ldots \circ p_{\alpha-\mu}$ suffices to put P into orientation α_{e_1}.

3.4 Effect of Shape Tolerance on the Push Function

When part geometry changes, it affects the push function in two ways. First, it will change the value of α_{l_i} and α_{r_i}, that is, it will change the length of one or more open half intervals. Second, some critical change on the value of α_{l_i} and α_{r_i} may cause a stable edge in the nominal part to become unstable. Also, it may cause an unstable edge to become stable. Since the number of steps of the push function is equal to the number of stable edges, this will combine several half intervals into a bigger one or break one half interval into several smaller ones. We refer to this as a *topological change* of the push function. It may dramatically change the ranking of the $|l(e_i)|$'s. An example of a polygon with tight topological constraints is shown in Figure 5. e_1 is the stable edge with the largest α_{l_i}. However, if the coordinates of v_1 undergo a small change, it may cause e_1 to become unstable. A possible topological change of the push function for this polygon is shown in Figure 6.

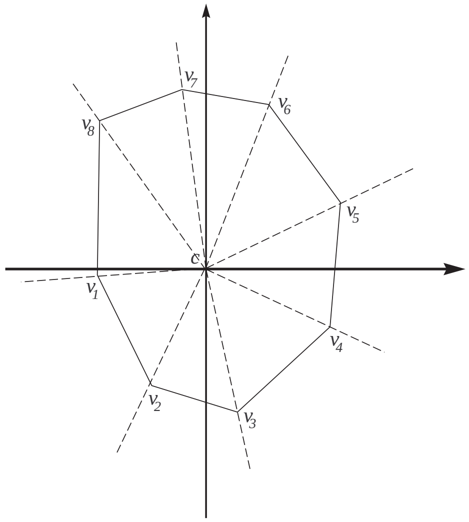

Figure 5: *Example of a polygon with tight topological constraint.*

Using Chen and Ierardi's algorithm, we can find a push plan that is valid for all the parts in the tolerance class $\Delta(P, \varepsilon)$ if

- for every part in $\Delta(P, \varepsilon)$, e_1 is the stable edge with

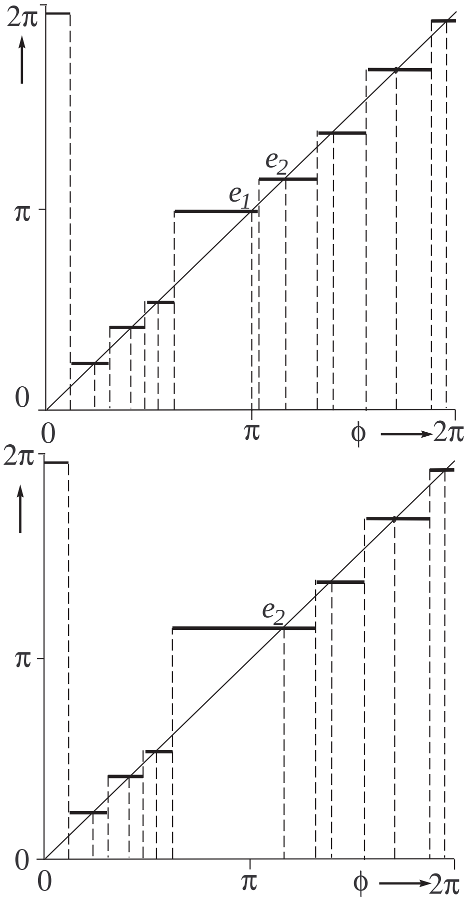

Figure 6: *Topological change of the push function for the part in Figure 5when v_1 is moved.*

the largest left half interval, and

- if each member $P' \in \Delta(P, \varepsilon)$ is characterized by a pushing angle set Ψ', and the intersection of all Ψ' is nonempty.

We can state this differently. If the lower bound of $|l(e_1')|$ for any part $P' \in \Delta(P, \varepsilon)$ is larger than the upper bound of all other left half intervals, then there will be a push plan which will work for all the parts in the tolerance class.

3.5 Algorithm to Compute ε

We now compute the radius of a tolerance class for which a common push angle exists. The algorithm takes as input the geometry of the nominal part and computes the largest ε such that there exists a plan to feed all the parts in $\Delta(P, \varepsilon)$, and a push angle α that defines the plan.

Initially, we assume that there is no topological change on the push function within the tolerance class $\Delta(P, \varepsilon)$, and study the relationship between $|l(e_i')|$ and ε where e_i is a stable edge. Afterward, we address the issue of topological changes in the push function.

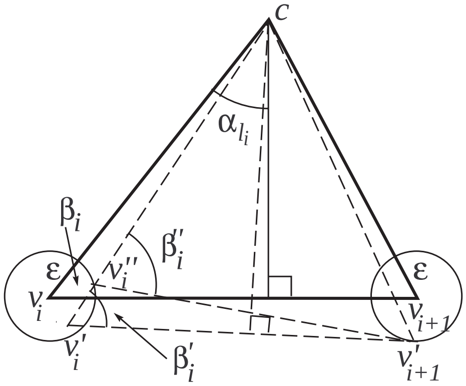

Figure 7: *Case 1. v_i is an equilibrium vertex thus $l(e_i) = \alpha_{l_i}$.*

There are two cases we need to consider, according to whether v_i is an equilibrium vertex or not.

Case 1. As shown in Figure 7, v_i is an equilibrium vertex, thus $l(e_i) = \alpha_{l_i}$. Let $\beta_i = \pi/2 - \alpha_{l_i}$. First we derive a lower bound on $l(e_i)$. We start by making the following observations:

1. Minimizing α_{l_i} is equivalent to maximizing β_i. As shown in Figure 7 and by definition $\alpha_{l_i} + \beta_i = \pi/2$.
2. For every v_i' within the ε disk around v_i, the maximum β_i is achieved when e_i' is tangent to the ε disk around v_{i+1}. So given the vertex v_i', the maximum value of α among all $P' \in \Delta(P, \varepsilon)$ and $v_i' \in P'$ is well defined. The problem now becomes to find the point v_i' that can maximize β_i.
3. For any v_i' *inside* the ε disk around v_i, there is a point v_i'' *on* the ε disk around v_i such that $\beta_i'' > \beta_i$. Thus we need only consider points on the boundary of the circular tolerance zone.

In P. Agarwal, L. Kavraki, M. Mason, editors, Robotics: The Algorithmic Perspective, A. K. Peters, Ltd. 1998.

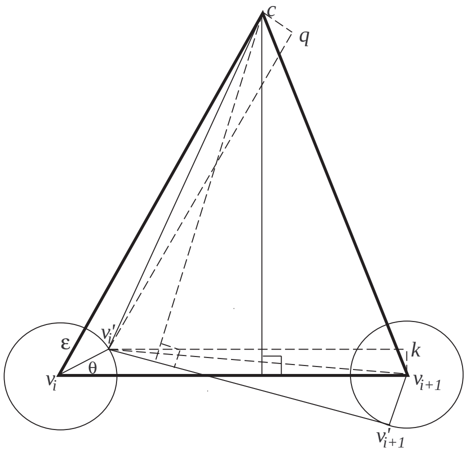

Figure 8: *Relationship between* $\min(|l(e_i')|)$ *and* ε *in case 1.* $\angle cv_i'q = \gamma$, $\angle kv_i'v_{i+1} = \lambda$, $\angle v_{i+1}v_i'v_{i+1}' = \delta$.

A point v_i' on the circle around v_i can be represented by a single variable θ. $v_i'v_{i+1}'$ is tangent to the ε disk around v_{i+1} since this maximizes β_i' when we fix v_i'. To derive the relationship between β_i' and ε, we draw a straight line $v_i'q$ parallel to v_ic with cq perpendicular to $v_i'q$ and $v_i'k$ parallel to v_iv_{i+1} with $v_{i+1}k$ perpendicular to $v_i'k$. Let $\angle cv_i'q = \gamma$, $\angle kv_i'v_{i+1} = \lambda$, $\angle v_{i+1}v_i'v_{i+1}' = \delta$. We have:

$$\beta_i' = \gamma + \beta_i + \delta + \lambda \tag{1}$$

Lemma 1 *When we only consider points on the* ε *ball, the relationship between* $\min(|l(e_i')|)$ *and* ε *can be expressed as*

$$\begin{aligned}\min(|l(e_i')|) &= |l(e_i)| \\ &- \arcsin \frac{\varepsilon \sin(\beta_i - \theta)}{\sqrt{(|cv_i| - \varepsilon\cos(\beta_i - \theta))^2 + \varepsilon^2 \sin^2(\beta_i - \theta)}} \\ &- \arcsin \frac{\varepsilon(1+\sin\theta)}{\sqrt{(|e_i| - \varepsilon\cos\theta)^2 + \varepsilon^2\sin^2\theta}},\end{aligned} \tag{2}$$

where θ *is the value in* $[0, 2\pi)$ *that minimizes the right-hand side.*

Proof. From Figure 8:

$$\begin{aligned}\gamma &= \arcsin \frac{|cq|}{|cv_i'|} \\ &= \arcsin \frac{\varepsilon \sin(\beta_i - \theta)}{\sqrt{(|cv_i| - \varepsilon\cos(\beta_i - \theta))^2 + \varepsilon^2 \sin^2(\beta_i - \theta)}}\end{aligned}$$

$$\delta = \arcsin \frac{\varepsilon}{|v_i'v_{i+1}'|} = \arcsin \frac{\varepsilon}{\sqrt{(|e_i| - \varepsilon\cos\theta)^2 + \varepsilon^2\sin^2\theta}}$$

$$\lambda = \arcsin \frac{\varepsilon\sin\theta}{|v_i'v_{i+1}'|} = \arcsin \frac{\varepsilon\sin\theta}{\sqrt{(|e_i| - \varepsilon\cos\theta)^2 + \varepsilon^2\sin^2\theta}}$$

For any given ε, we can find the value of θ that maximize $\gamma + \delta + \lambda$. Using Equation 1, the relationship between $\min(|l(e_i')|)$ and ε as stated in the lemma follows. □

We can then show

Lemma 2 *When we only consider points on the* ε *ball, the relationship between* $\max(|l(e_i')|)$ *and* ε *can be expressed as*

$$\begin{aligned}\max(|l(e_i')|) &= |l(e_i)| \\ &+ \arcsin \frac{\varepsilon \sin(\beta_i - \theta)}{\sqrt{(|cv_i| - \varepsilon\cos(\beta_i - \theta))^2 + \varepsilon^2 \sin^2(\beta_i - \theta)}} \\ &+ \arcsin \frac{\varepsilon(1+\sin\theta)}{\sqrt{(|e_i| - \varepsilon\cos\theta)^2 + \varepsilon^2\sin^2\theta}},\end{aligned} \tag{3}$$

where θ *is the value in* $[0, 2\pi)$ *that maximizes the right-hand side.*

Case 2. As shown in Figure 9, v_i is *not* an equilibrium vertex. Thus $l(e_i)$ is defined by α_{e_i} and α_{v_j}, where v_j is the last equilibrium vertex ($j \neq i$) (namely, when we proceed clockwise from v_i along the boundary of P v_j is the first equilibrium vertex that we encounter).

In this case, we argue the following

Lemma 3 *When we only consider points on the* ε *ball, the relationship between* $|l(e_i')|$ *and* ε *can be expressed as*

$$\begin{cases}\min(|l(e_i')|) = |l(e_i)| - \arctan\frac{\varepsilon}{|cv_j|} - \arctan\frac{2\varepsilon}{|e_i|} \\ \max(|l(e_i')|) = |l(e_i)| + \arctan\frac{\varepsilon}{|cv_j|} + \arctan\frac{2\varepsilon}{|e_i|}\end{cases} \tag{4}$$

Proof. We observe that $|l(e_i')| = |l(e_i)| + \alpha_{v_j} - \alpha_{v_j'} + \alpha_{e_i'} - \alpha_{e_i}$.

In P. Agarwal, L. Kavraki, M. Mason, editors, Robotics: The Algorithmic Perspective, A. K. Peters, Ltd. 1998.

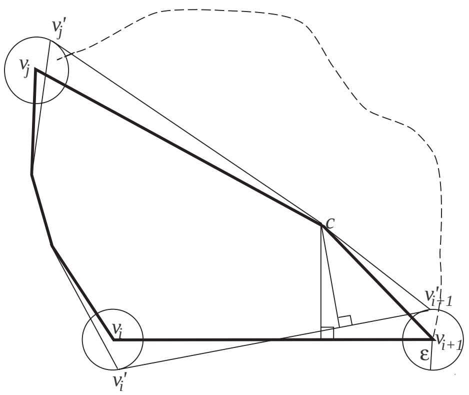

Figure 9: *Case 2. v_2 is not an equilibrium vertex thus $l(e_i) = \alpha_{e_i} - \alpha_{v_j}$ where v_j is the last equilibrium vertex $(j \neq i)$.*

Since $-\arctan \frac{\varepsilon}{|cv_j|} \leq \alpha_{v_j} - \alpha_{v_j'} \leq \arctan \frac{\varepsilon}{|cv_j|}$ and $-\arctan \frac{2\varepsilon}{|e_i|} \leq \alpha_{e_i'} - \alpha_{e_i} \leq \arctan \frac{2\varepsilon}{|e_i|}$, the relationship between $|l(e_i')|$ and ε as stated in the lemma follows. □

Combining both cases, we find that for every stable edge e_i, if there is no topological change on the push function, we can express the relationship between $|l(e_i')|$ and ε as Equation 2, 3 or 4. They form a set of nonlinear constraints. From these constraints, we can compute the largest ε zone numerically in which $\min(|l(e_1)| \geq \max(|l(e_i)|$ for all $i \neq 1$ and e_i is a stable edge.

Now we consider topological changes on the push function. As we discussed before, it is caused by critical change on the value of α_{l_i} and α_{r_i}. Since in Case 1 $|l(e_i)| = \alpha_{l_i}$, we already know the relationship between α_i and ε. Each possible critical change on the value of α_{l_i} and α_{r_i} can be expressed as a linear constraint on ε. Given ε, we can determine which topological changes of the push function can occur. Thus we can first ignore the topological constraints, and compute ε. Then we check whether ε is valid for all the topologically different situations that are possible within $\Delta(P, \varepsilon)$. If not, we decrease ε to exclude the first topological class which violates the tolerance condition. We repeat until no more violations occur.

Before considering topological constraints, the algorithm runs in time $O(n)$. Since there are $O(n)$ topological constraints and each takes $O(n)$ time to handle, the overall complexity for computing ε is $O(n^2)$.

3.6 Convexity

Intuition suggests that if the condition above holds for all extrema in the tolerance zone, then the condition will hold for any part in the tolerance class. Unfortunately, this is not the case for feeding. We construct a counterexample, illustrated in Figure 10, where the size of the left half interval shrinks, then grows as we move a vertex continuously from v_{i+1} to $v^{''}_{i+1}$.

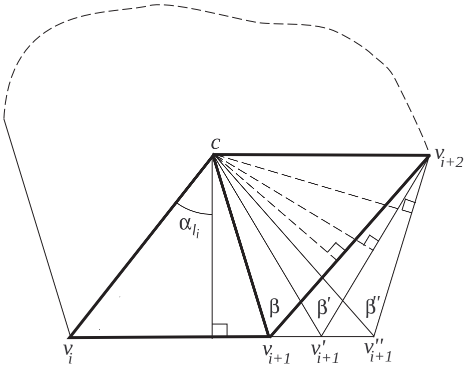

Figure 10: *Counterexample where convexity does not hold.*

Let c be the COM of a nominal part with adjacent edges e_i and e_{i+1} defined by v_i, v_{i+1}, v_{i+2} respectively. We construct cv_{i+2} such that it is parallel to $v_i v_{i+1}$ and $|v_{i+1}v_{i+2}| > |cv_{i+1}|$. Next, extend $v_i v_{i+1}$ to $v^{''}_{i+1}$ such that $|cv^{''}_{i+1}| = |v_{i+1}v_{i+2}|$ and $|cv_{i+1}| = |v^{''}_{i+1}v_{i+2}|$. Let $v^{'}_{i+1}$ be the midpoint of $v_{i+1}v^{''}_{i+1}$. Let $l(e_{i+1})$, $l(e^{'}_{i+1})$ and $l(e^{''}_{i+1})$ be the left environment of the corresponding edge. Since $|l(e_{i+1})|+\beta = \pi/2$, $|l(e^{'}_{i+1})|+\beta^{'} = \pi/2$, $|l(e^{''}_{i+1})| + \beta^{''} = \pi/2$ and $\beta^{'} > \beta = \beta^{''}$, then $|l(e^{'}_{i+1})| < |l(e_{i+1})| = |l(e^{''}_{i+1})|$. Thus, when we move vertex v_{i+1} to $v^{'}_{i+1}$, $|l(e_{i+1})|$ initially shrinks, then grows back to its original size.

For a polygon where $|l(e_{i+1})|$ is the critical half interval, the same push plan will work for v_{i+1} and $v^{''}_{i+1}$ but may not work for $v^{'}_{i+1}$. This demonstrates that, in the context of feeding, we cannot naively assume convexity and test only extrema in the tolerance zone.

In P. Agarwal, L. Kavraki, M. Mason, editors, Robotics: The Algorithmic Perspective, A. K. Peters, Ltd. 1998.

3.7 Implementation and Experimental Results

We implemented the algorithm and ran it on 9 nominal parts, six of which are shown in Figure 11.

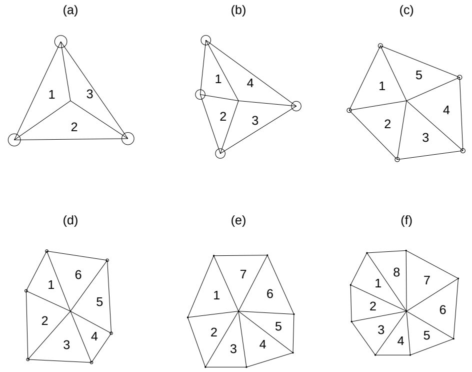

Figure 11: *Nominal polygons and their tolerance zones. The numbers indicate part edges.*

To test that the resulting tolerance zones were correct, we generated 1000 random variations for each part by allowing vertices to vary uniformly within each tolerance zone. As expected, the nominal plan worked for all parts in the tolerance class. Zero failures are indicated in the second column in Table 1.

We then increased the tolerance zone by a factor of $1.1, 1.2, 1.3, 1.4$ respectively, and in each case generated 1000 random variations as above. Here, as expected, the nominal plan sometimes fails since some polygons lie outside the tolerance zone. Although the percentage is small, this numerical experiment suggests that the tolerance zone we compute provides a tight bound.

Small changes in part geometry can cause large changes in the tolerance class. For example, the two 8-sided parts in Figure 12 are identical except for the y coordinate of vertex v_1, which in (a) is -0.1 and in (b) is -2.3. Note that the radius of the tolerance class in (b) is significantly larger than in (a)! This is because the part in (a) has a tight topological constraint. This suggests that if we change the design of part from (a) to (b), it will be much more robust to variations in part shape.

(# of edges)	1.0ε	1.1ε	1.2ε	1.3ε	1.4ε
3	0	0.7	1.2	3.9	5.8
4	0	0.2	0.9	2.1	3.0
5	0	0.7	1.8	4.3	5.6
6	0	0.3	0.9	3.6	5.2
7	0	0.0	0.5	2.1	3.0
8	0	0.2	1.1	2.5	6.0
9	0	0.3	0.9	2.1	5.0
10	0	0.4	1.4	2.4	5.3
17	0	0.1	0.5	1.7	3.0

Table 1: *Results for 9 nominal parts, 5000 trials each. For each part and tolerance radius, the table lists the percentage of polygons that cannot be fed with the nominal plan. The first column corresponds to polygons in the computed tolerance class (no failures). The other columns are polygons outside the tolerance class.*

4 Shape Tolerance in Fixturing

Wentink, van der Stappen and Overmars [2] show that any convex polygon without parallel edges can be immobilized by a right angle fixel and a single point contact provided with a clamp. Let P be a convex polygon. When placed in the fixture, it has a horizontal edge contact and a vertical point contact on the right angle fixel. We assume that the contacts are frictionless. This is a conservative assumption since any fixture computed assuming zero friction also holds in presence of friction. We label the two endpoints of the edge contact as v_1 and v_2, the point contact as v_p, and the vertices of the edge to which we apply the clamp as v_{c1} and v_{c2}. Edge $v_{c1}v_{c2}$ is called the *clamp edge*; see Figure 13 for an illustration. If we draw vertical lines from v_1 and v_2, they intersect the horizontal line from v_p at points f_1 and f_2 respectively. We refer to the line segment f_1f_2 as the *fixturing segment* since the condition for the right angle fixel and clamp to immobilize the part is that the normal of the clamp edge at the clamp contact point intersects the fixturing segment. For details see [2].

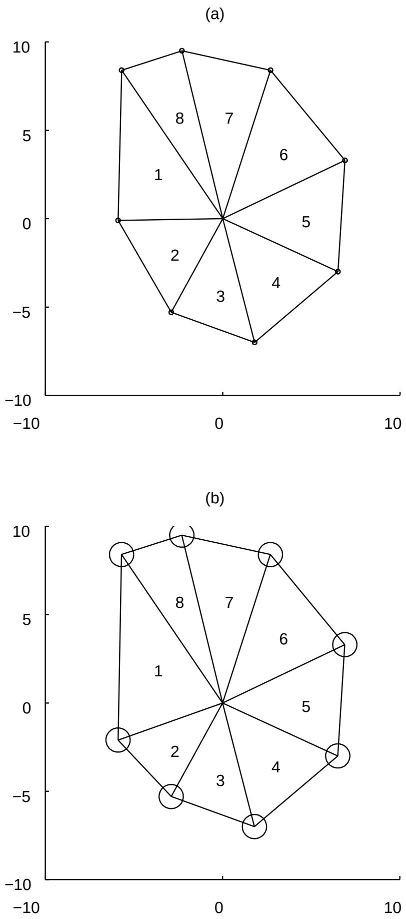

Figure 12: *Computed tolerance zones for two similar polygons.*

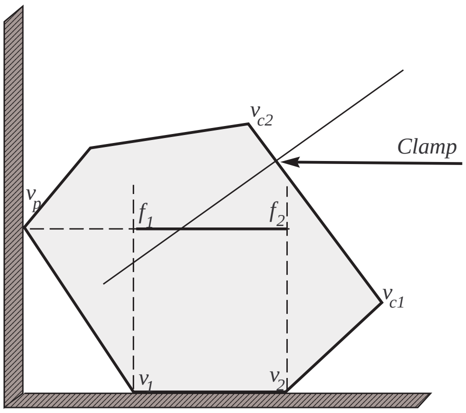

Figure 13: *Fixturing with a right angle fixel and a point clamp.*

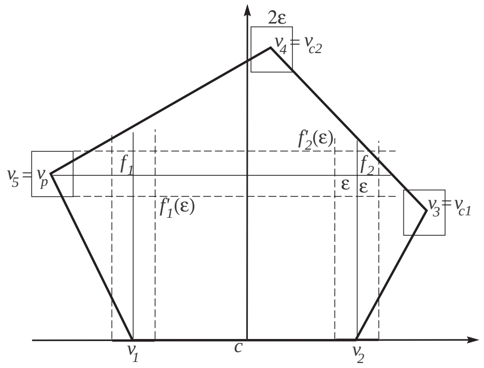

Figure 14: *Description of the part and its shape uncertainty in fixturing.*

4.1 A Tolerance Class for Fixturing

As before, let P be a convex polygonal part. Its n vertices are counterclockwise labeled $v_1 \dots v_n$. Edge $v_1 v_2$ is the edge that we place on the horizontal arm of the right-angle fixel. We assume that a fixed coordinate frame is attached to P. We choose the midpoint of the edge $v_1 v_2$ as the origin which we denote by c, and align the x-axis with the edge $v_1 v_2$. All the coordinates are expressed relative to this frame (Figure 14).

Now we define the parametric tolerance class $\Delta(P, \varepsilon)$. Again, we assume that part edges are always straight. For fixturing, we replace the circles around vertices with boxes and line segments. This is more natural for fixturing because horizontal and vertical shifts play a more prominent role than rotation. Let P be the nominal part, and let P' be a candidate. Let c' denote the center of the edge $v_1' v_2'$ of P'. We say that $P' \in \Delta(P, \varepsilon)$ if we can place part P' such that c' is coincident with c, the bottom edges are aligned, and all the vertices of P' are within a square of size ε around the corresponding vertices of the nominal part P. It is easy to check if a candidate $P' \in \Delta(P, \varepsilon)$ in $O(n)$ time. An example is shown in Figure 14.

In P. Agarwal, L. Kavraki, M. Mason, editors, Robotics: The Algorithmic Perspective, A. K. Peters, Ltd. 1998.

4.2 Effect of Shape Variations

A fixture design with a right angle fixel is described by the edge of the part that must touch the bottom arm of the fixel, and the height of the clamp. A part P' is loaded into the fixture design by placing it with edge $v_1 v_2$ on the horizontal arm, shifting it to the left until it hits the vertical arm, and shifting the clamp horizontally until it touches the part. A part P' is said to be fixtured by a particular fixture design, if, after loading it, it is indeed immobilized. A fixture design is considered valid for the tolerance part set $\Delta(P, \varepsilon)$ if it fixtures each part in the set and the clamp will touch the part at the same edge for each part.

When the part shape changes in the tolerance class, the fixturing segment may change in height and length, the clamp contact point may move horizontally and its normal may rotate. Also, the vertex v_p that makes the contact at the vertical arm of the fixel may be changed. In this paper, we restrict our tolerance zone such that this last type of change does not happen and we only consider the case where v_1, v_2, v_p, v_{c1}, and v_{c2} are distinct vertices.

4.3 Convexity

For fixturing as well we show that convexity does not necessarily help in analyzing the tolerance zone. Namely, we show that it is insufficient to analyze a fixture at the extrema of a tolerance zone, as its behavior can change between extrema.

Assume that we have two polygons P and P'' that are the same except for the location of one vertex which is at position A in P and at position A'' in P''. Now assume both P and P'' can be held with the same fixture. Convexity would suggest that as we move the vertex from A to A'' along the line segment AA'', the resulting polygons will also be held with the same fixture. If convexity holds, then we only need to consider combinations of all the vertices of the tolerance boxes to check if for a certain ε a valid fixture design exists. Unfortunately, this is not always true, as the following counterexample shows (see Figure 15).

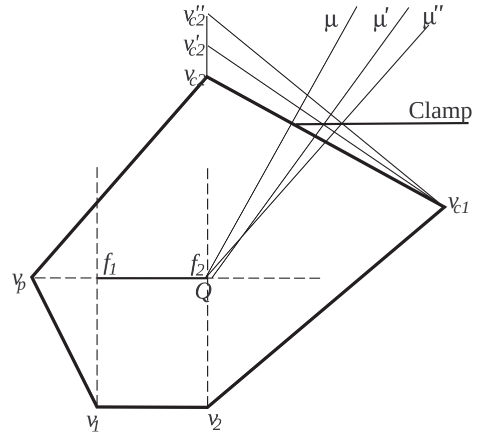

Figure 15: *A counterexample: convexity is violated.*

As illustrated in the figure, we move the vertex v_{c2} to v''_{c2}. We denote by μ and μ'' the normals at the clamp contact point of edge $v_{c1}v_{c2}$ and $v_{c1}v''_{c2}$ respectively. The two normals intersect at point Q. It is easy to construct a polygon such that Q is on the fixturing segment. Since μ and μ'' both intersect the fixturing segment at Q, the monotonicity should imply that for every point v'_{c2} on the line segment $v_{c2}v''_{c2}$, the normal μ' must also intersect the fixturing segment at Q. (Otherwise it is easy to construct a polygon such that the fixturing segment is so small that it does contain Q but not the other intersection point.) But this is obviously false. So, we have shown that the x-coordinate of the intersection of the normal at the clamp contact point and the fixture line segment does not always change monotonically. Thus a valid fixture for polygons with vertex at v_{c2} and v''_{c2} may not be valid for every polygon with its vertex on $v_{c2}v''_{c2}$.

4.4 An Algorithm to Compute max ε

Since we assume that v_1,v_2,v_p,v_{c1} and v_{c2} are distinct vertices, a fixture is valid for $\Delta(P, \varepsilon)$ if and only if for every $P' \in \Delta(P, \varepsilon)$, the normal at the clamp lies between $f_1(\varepsilon)$ and $f_2(\varepsilon)$ (see Figure 14). Now consider the edge at which we apply the clamp. The normal at the clamp point will intersect the fixture segment at some position x. We next show that the toleranced edge that achieves the maximum (or minimum) x must be such that at least one endpoint is coincident with one of four extreme points that define the two tolerance boxes at its endpoints. Consider Figure 16. This demonstrates graphically that for any edge l_2 in the tolerance zone which does not have an endpoint at one of the four extreme points ($ABCD$). We can always move the edge parallel to itself until one of the endpoint reaches the boundary, and get a larger (smaller) x. So we have shown that the maximum or minimum of x will be achieved only when at least one endpoint of the edge is coincident with one of the four extreme points ($ABCD$). Note that, as we have shown before, it is not necessary that both endpoints are extreme points.

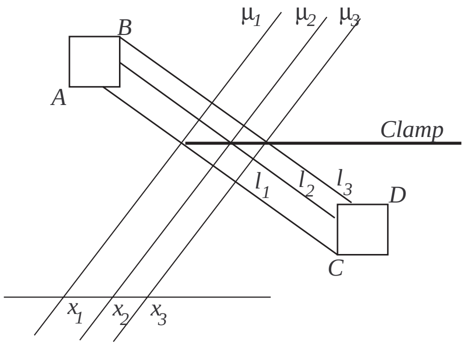

Figure 16: *Boundary condition.*

Now let us fix one of the endpoints of the edge and look at the relationship between the orientation of the clamp edge and the x-coordinate of intersection of the normal and the fixture segment. As indicated in Figure 17, let y be the vertical distance from the fixed endpoint to the fixture segment, and let h be the vertical distance from the clamp to the fixture segment. Let x_1 be the x-coordinate of the fixed endpoint, x_2 the x-coordinate of the clamp contact point, and x_3 the x-coordinate of intersection between the normal and the fixture segment. Finally, let θ be the orientation of the clamp edge.

Because the normal is perpendicular to the clamp edge, we have

$$\frac{y-h}{x_2-x_2} = \frac{x_2-x_3}{h} = \tan\theta \qquad (5)$$

From Equation 5 , we eliminate x_2 and get

$$x_3 = x_1 + (y-h)\cot\theta - h\tan\theta \qquad (6)$$

Take the derivative of x_3 on θ and set it to be 0,

$$-\frac{y-h}{\sin^2\theta} - \frac{h}{\cos^2\theta} = 0 \qquad (7)$$

Solving Equation 7, we get

$$\theta = \arctan\sqrt{1-\frac{y}{h}} \qquad (8)$$

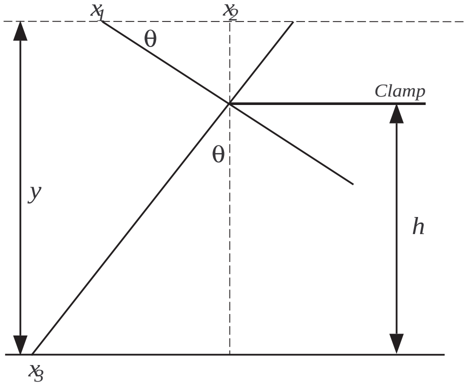

Figure 17: *Relationship between θ and x_3 when we fix one endpoint of the clamp edge.*

Equation 8 has at most one solution in $[0, \pi/2]$. So, given ε, for each fixed endpoint we only need to consider at most three directions: the maximal and minimal θ within the tolerance zone and the solution of Equation 8. Also we only need to consider four fixed endpoints (the corners of the tolerance box) at each side of the clamp edge. Looking at this carefully, we get a total of 8 clamp edges and 2 points ($f_1(\varepsilon)$ and $f_2(\varepsilon)$) within the tolerance zones that might define the maximum or minimum position of the intersection of the normal and the fixture edge. (The 8 edges are the four tangent lines of the two tolerance boxes plus, for each of the 4 extreme vertices (2 per box), the angle that comes out of Equation 8.) This leads to a system with a constant number of equations from which we can compute the largest ε in which the fixture design will work for all parts in $\Delta(P, \varepsilon)$ (assuming no topological change). Since only five vertices are relevant in the calculation of ε, the complexity of this algorithm is $O(1)$.

In P. Agarwal, L. Kavraki, M. Mason, editors, Robotics: The Algorithmic Perspective, A. K. Peters, Ltd. 1998.

4.5 Implementation and Experimental Results

We implemented the above algorithm and ran similar test as we did in feeding on 4 nominal parts(as shown in Figure 18, the results are listed in Table 2. The numerical experiment suggests that the tolerance zone we compute provides a tight bound.

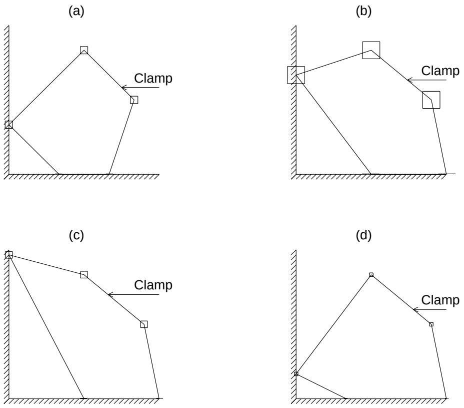

Figure 18: *Nominal polygons and their tolerance zones.*

Polygon	1.0ε	1.1ε	1.2ε	1.3ε	1.4ε
a	0	0.1	0.1	0.6	1.3
b	0	0.1	0.3	1.3	2.9
c	0	0.2	0.2	0.8	1.9
d	0	0.0	0.1	0.4	1.4

Table 2: *Results for 4 nominal parts, 5000 trials each. For each part and tolerance radius, the table lists the percentage of polygons that cannot be immobilized by the nominal plan. The first column corresponds to polygons in the computed tolerance class (no failures). The other columns are polygons outside the tolerance class.*

The same part may have very different tolerance zones for different fixture plans. For example, the two parts in Figure 19 are identical, but the clamp height is different. Note that the radius of the tolerance class in (b) is significantly larger than in (a)! This suggests that if we change the fixture plan from (a) to (b), it will be much more robust to variations in part shape.

5 Conclusion and Future Work

In general, it is very difficult to characterize the range of mechanical behavior for the uncountable set of parts in a tolerance class. This is especially true for geometric tolerancing, where there is no limit on the part edges within the zone. For example, it is easy to see that a part that meets the geometric zone specification can wind up in a broad range of final orientations when pushed with a fence. It is not at all clear how to bound this range.

Recently there has been renewed interest in parametric tolerance classes and associated algorithms. Fast checking algorithms are very useful during the design cycle. Ideally, part geometry can be monitored in real time as is currently done with VLSI CAD design rule checking.

This paper contributes by rigorously characterizing two new parametric tolerance classes. For each, we give an $O(n)$ time algorithm for testing if an n-sided part is in the class. For feeding we give an $O(n^2)$ algorithm to compute the radius of the largest allowable tolerance zone around each vertex. For fixturing we give an $O(1)$ algorithm to compute the dimensions of rectangular tolerance zones. We implemented both algorithms and illustrate with experimental results.

In both cases the tolerance class is defined with respect to the context. For feeding, since the center of mass plays an important role in fence mechanics, it was very helpful to define variations in shape with respect to a coordinate frame attached to the center of mass. Similarly for fixturing, where variations in shape are relative to the part edge that is in contact with the edge fixel. This suggests an alternative to defining shape tolerance in the abstract.

Efficient algorithms are necessary to provide rapid feedback to designers. As the examples in Figures 12 and 19 indicate, small changes in part or fixture geometry can imply large changes in tolerance sensitivity. In a Computer Aided Design system, low complexity algorithms such as those described in this paper can run in the background to rapidly indicate tolerance effects. Future research is required to anticipate and suggest

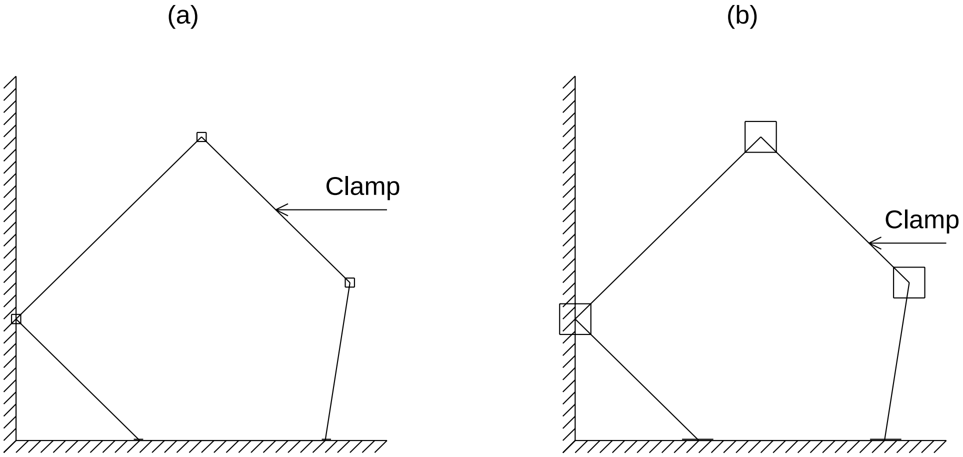

Figure 19: *Shape tolerance computation for fixturing. Intuition suggests that the clamp position in (a) would be more robust to shape variation than the clamp position (slightly lower) in (b). However, the algorithm computes a smaller allowable epsilon box in (a) than in (b). This illustrates that small changes in fixture design can produce counterintuitive effects on shape tolerance.*

changes in part geometry that can improve tolerance sensitivity.

Last, the issue of convexity arose repeatedly in this study. We found that intuition often failed us: we guessed that if a solution exists at all extrema, it would hold everywhere in between. To our surprise, we discovered counterexamples. Our results and methodology for constructing counterexamples may be applicable in other contexts such as assembly and motion planning.

Acknowledgements

This work was supported in part by NSF Awards IRI-9612491 and Presidential Faculty Fellow Award IRI-9553197 to Prof. Goldberg and by NATO Travel Grant CRG 951224 (joint with M. Overmars).

Work on this paper by D. Halperin has been supported in part by an Alon Fellowship, by ESPRIT IV LTR Project No. 21957 (CGAL), by the USA-Israel Binational Science Foundation, by The Israel Science Foundation founded by the Israel Academy of Sciences and Humanities, and by the Hermann Minkowski – Minerva Center for Geometry at Tel Aviv University.

References

[1] R. P. Berrety, M. Overmars, F. Van der Stappen, and K. Goldberg. On fence design and the complexity of push plans for orienting parts. In *13th Symposium on Computational Geometry*, Nice, France, June 1997. ACM.

[2] M. Overmars, C. Wentink, and Frank van der Stappen. Fixture planning. In *Second Workshop on Algorithmic Foundations of Robotics*, Toulouse, France, July 1996.

[3] James L. Nevins and Daniel E. Whitney. Computer-controlled assembly. *Scientific American*, 1978.

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

[5] Ken Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10(2):201–225, August 1993. Special Issue on Computational Robotics.

[6] Anil Rao and Ken Goldberg. Manipulating algebraic parts in the plane. *IEEE Transactions on Robotics and Automation*, 11(4), August 1995.

[7] Yui-Bin Chen and Doug Ierardi. The complexity of oblivious plans for orienting and distinguishing polygonal parts. *Algorithmica*, 14, 1995.

[8] A. A. G. Requicha. Toward a theory of geometric tolerancing. *IJRR*, 2(4), 1983.

[9] U. Roy, C. Liu, and T. Woo. Review of dimensioning and tolerancing: Representation and processing. *Computer-Aided Design*, 23(7), 1991.

[10] H. Voelker. A current perspective on tolerancing and metrology. *Manufacturing Review*, 6(4), 1993.

[11] L. Joskowicz, E. Sacks, and V. Srinivasan. Kinematic tolerance analysis. In *Second Workshop on Algorithmic Foundations of Robotics*, Toulouse, France, July 1996.

[12] C. Yap and Ee-Chien Chang. Geometric tolerancing: Theory, issues, and computation. In *Second Workshop on Algorithmic Foundations of Robotics*, Toulouse, France, July 1996.

[13] J. C. Latombe, R. H. Wilson, and F. Cazals. Assembly sequencing with toleranced parts. In *Third Symposium on Solid Modelling and Applications*, 1995.

[14] S. Akella. *Robotic Manipulation for Parts Transfer and Orienting: Mechanics, Planning, and Shape Uncertainty.* PhD thesis, The Robotics Institute, Carnegie Mellon University, December 1996. Robotics Institute Technical Report CMU-RI-TR-96-38.

[15] S. Akella and M. T. Mason. Parts orienting with shape uncertainty. In *Submitted to ICRA 98*, October 1997.

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

[17] R. C. Brost and R. R. Peters. Automatic design of 3d fixtures and assembly pallets. In *IEEE International Conference on Robotics and Automation*, 1996.

[18] Lydia Kavraki. Part orientation with programmable vector fields: Two stable equilibria for most parts. In *IEEE International Conference on Robotics and Automation*, Albuquerque, New Mexico, April 1997.

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

[20] S. Akella, W. Huang, K. Lynch, and M. Mason. Planar manipulation on a conveyor by a one joint robot with and without sensing. In *Second Workshop on Algorithmic Foundations of Robotics*, Toulouse, France, July 1996.

[21] J. Wiegley, K. Goldberg, M. Peshkin, and M. Brokowski. A complete algorithm for designing passive fences to orient parts. *Assembly Automation*, 17(2), August 1997.

[22] Matthew T. Mason. Mechanics and planning of manipulator pushing operations. *International Journal of Robotics Research*, 5(3), Fall 1986.

In P. Agarwal, L. Kavraki, M. Mason, editors, Robotics: The Algorithmic Perspective, A. K. Peters, Ltd. 1998.