

# Minimal Trap Design

Pankaj K. Agarwal\*, Anne D. Collins† and John L. Harer‡

## Abstract

This paper addresses the issue of trap design for sensorless automated assembly. First, we present a simple algorithm that determines in  $O(nm \alpha(nm) \log(nm))$  time whether an  $n$ -sided polygonal part will fall through an  $m$ -sided polygonal trap. We then introduce the notion of a *minimal trap* for a polygonal part, and develop an algorithm to design a family of minimal feeders built from these traps. The algorithm runs in  $O(k n^{3+\epsilon})$  time, where  $k$  is the number of stable orientations of  $P$ . Moreover, it is complete in the sense that we can always find a feeder, provided that one exists that rejects and supports the appropriate poses of the part.

## 1 Introduction

A *feeder* is a mechanism that takes a collection of parts in arbitrary orientations, and returns a stream of identical parts all oriented the same way. One of the most commonly used mechanisms for sensorless part feeding is the *vibratory bowl feeder*, which works as follows: a helical track starting at the bottom of a bowl circles the inside wall as it climbs to the top, and the bowl is set to vibrate in such a way that a part lying on the track is forced to move up. (We ignore the actual mechanics of how this happens, and pretend the motion is smooth; see [4].) Along this track is placed a sequence of obstacles – ramps, wiper blades, traps, etc. – designed to allow only parts in the desired orientation to reach the top; the rest fall back into the bowl to start again at the bottom. The obsta-

cles we consider are *traps*, collections of strategically placed holes cut into the track through which undesirable orientations of the part will fall. Researchers have used various techniques including genetic algorithms [5], heuristics [7], and geometric algorithms [3] to design trap-based feeders; see [2] for a detailed survey on this topic. In this paper, we adopt the geometric approach that was first developed in [3].

In Section 2, we describe the model of the bowl feeder that we use. In Section 3, we propose a new algorithm to determine whether a trap drops a part. Our algorithm relies on a considerably simpler data structure than the existing  $O((nm(n+m))^{1+\epsilon})$  algorithm [3],<sup>1</sup> reducing the time to  $O(nm \alpha(nm) \log(nm))$ . In Section 4, we associate with a part a new class of traps which are *minimal* in the sense that any trap that drops the part must contain one of them. This in turn leads to an algorithm for designing a family of feeders in  $O(k n^{3+\epsilon})$  time, under a model of computation in which the roots of a fixed degree polynomial can be computed in  $O(1)$  time. Here,  $k$  is the number of stable orientations of the part. Each feeder supports one of these orientations and drops all others; moreover, the family is complete, in the sense that if a feeder exists that supports one pose and drops the rest, we will find one. Finally, we discuss a modification of our minimal traps, through which the corresponding part will actually fall. If the polygons are convex, we again obtain a complete family of feeders built from these modified minimal traps. Although this approach is no longer complete in the non-convex case, it nonetheless suggests a useful heuristic for feeder design.

## 2 Model

In this section we describe the geometric model of a vibratory bowl feeder. We assume that the bowl is so large relative to the size of the parts that the curvature of the track is negligible, and model the feeder as a straight track along which the parts slide. The track is tilted slightly towards a rail-

---

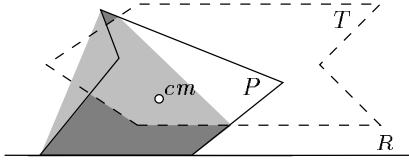
\*Department of Computer Science, Box 90129, Duke University, Durham, NC 27708-0129; pankaj@cs.duke.edu. Supported in part by National Science Foundation grants EIA-9870734, EIA-997287, and CCR-9732787, by Army Research Office MURI grant DAAH04-96-1-0013, by a Sloan fellowship, by a National Science Foundation NYI award and matching funds from Xerox Corporation, and by a grant from the U.S.-Israeli Binational Science Foundation.

†Department of Mathematics, Box 90320, Duke University, Durham, NC 27708-0320; annec@math.duke.edu. Supported in part by NSF VIGRE grant DMS-9983320 and NSF grant DMS-97-21428.

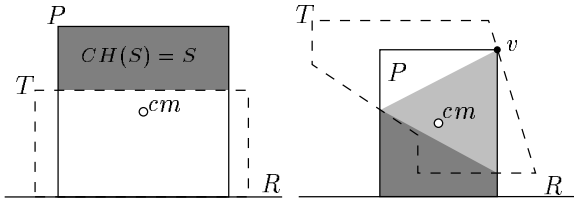
‡Department of Mathematics, Box 90320, Duke University, Durham, NC 27708-0320; John.Harer@duke.edu Supported in part by NSF grant DMS-97-21428.

---

<sup>1</sup>This has recently been improved to  $O(n^2 m \log n)$ ; see [2]



(a)  $cm \in CH(S) \Rightarrow P$  is supported.



(b) We assume that  $R$  does not support  $P$ , so this part falls into  $T$ .

(c)  $v \in \partial T$  is an isolated point of  $S$ . We assume this pose is supported.

Figure 1: Part  $P$  (solid) overlaps trap  $T$  (dashed). The small circle is  $P$ 's center of mass. The darkly shaded region is  $S$ , and the entire shaded region is  $CH(S)$ .

ing  $R$  that runs along one side; this ensures that parts remains in contact with  $R$  as they move. We only consider two-dimensional parts, so the entire system can be viewed in the plane.

Let  $P$  and  $T$  be simple polygons with  $n$  and  $m$  vertices, respectively.  $P$  represents the part. Due to the slight tilt in the track,  $P$  will assume one of  $O(n)$  stable orientations, as the center of mass seeks to be as low as possible; see [6]. The part therefore translates along the track with only one degree of freedom; we assume it moves with unit speed, and parameterize its position by time  $t$ .

The trap is a hole cut out of the track in the shape of polygon  $T$ . Note that, in a coordinate frame where the railing lies along the  $x$ -axis and the part translates in the  $(+x)$ -direction, any  $x$ -translation of  $T$  yields an equivalent trap; however,  $y$ -translations, rotations, and reflections do not. We set  $t = 0$  to be the time when the centers of mass of  $P$  and  $T$  have the same  $x$ -coordinate.

We assume that a point of  $P$  that lies over an edge of  $T$  is supported, with the exception of any edges of  $T$  along the railing. Thus the trap is technically  $T = \text{int}(T) \cup \text{int}(cl(T) \cap R)$ .<sup>2</sup> This allows

<sup>2</sup>Whether this assumption is reasonable depends upon the actual design of the bowl feeder. If instead we assume that  $R$  supports  $P$ , the trap would be  $\text{int}(T)$  and the part in Figure 1(b) would not fall. It is not difficult to modify our analysis for this situation.

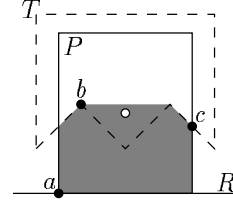


Figure 2:  $CH(S)$  has three types of vertices.

the part in Figure 1(b) to fall through the trap. For the remainder of the paper, we assume that all traps are of this form.

Denote by  $P(t)$  and  $cm(t)$  the part and its center of mass at time  $t$ . Define  $S(t)$ , the *supported region* of  $P$  at  $t$ , to be the portion of  $P$  that does not lie over the trap at time  $t$ :  $S(t) = P(t) - T$ ; see Figure 1. Note that  $S(t)$  may have multiple components, including isolated points (Figures 1(a),1(c)).

Define the part to be *safe* or *supported* at  $t$  if  $cm(t) \in CH(S(t))$ ; otherwise,  $P(t)$  is *rejected* or *dropped* at  $t$ . We say  $T$  drops  $P$  if  $P(t)$  is dropped by  $T$  for some time  $t$ , and  $P$  is supported by  $T$  otherwise. Note that “ $T$  drops  $P$ ” does not necessarily mean that the part actually falls through the trap; we defer this issue to Section 4.3.

Let  $V(t)$ , the *support vertices* of  $P(t)$  with respect to  $T$ , be the following subset of the vertices of  $S(t)$  (Figure 2): (a) vertices of  $P(t)$  outside the trap  $T$ , (b) reflex vertices of  $T$  contained in  $P(t)$ , and (c) intersections of an edge of  $P(t)$  and an edge of  $T$ . Then  $CH(S(t)) = CH(V(t))$ . We only consider reflex vertices of  $T$  since a convex vertex of  $T$  is a reflex vertex of  $S(t)$  and thus cannot be a vertex of  $CH(S(t))$ . In addition, if two edges intersect in an interval, we include only the endpoints in  $V$ ; thus,  $V(t)$  is always a discrete set of points. The support vertices vary continuously with  $t$  as  $P$  moves along the track, except at those times when a vertex in  $V(t)$  disappears or a new one appears. These events occur when a vertex of  $P(t)$  crosses the boundary of  $T$  or a reflex vertex of  $T$  crosses  $\partial P$ .

### 3 Decision Algorithm

In this section, we describe an algorithm for deciding whether a trap  $T$  drops an oriented part  $P$ . We also describe how to determine the set of *all* times when  $P$  drops,  $\{t \mid cm(t) \notin CH(S(t))\}$ .

The algorithm of Berretty et. al. [3] maintains  $CH(S(t))$  explicitly, by adapting the kinetic convex hull algorithm of [1]. Only some of this information is necessary, however; by maintaining a considerably simpler structure, we achieve a faster

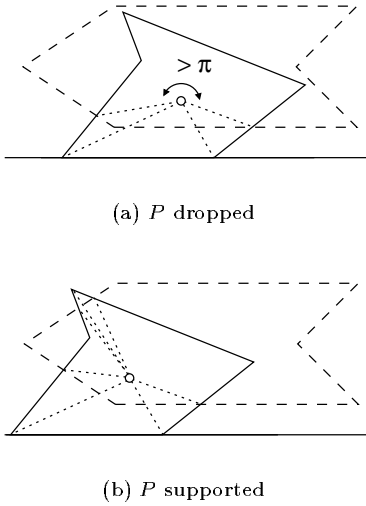


Figure 3: Whether  $P$  is dropped or supported depends upon the size of the largest empty cone about  $cm$ .

algorithm.

Our algorithm is based upon the following simple idea:  $T$  drops  $P(t)$  if and only if the angular measure of the largest cone centered at  $cm(t)$  that is empty of support vertices is greater than  $\pi$ ; see Figure 3. If such an empty cone exists, it must contain the ray from  $cm(t)$  in either the positive or negative  $x$ -direction.

We present two algorithms that find if and when the angular measure of the largest empty cone containing the  $(+x)$ -axis is greater than  $\pi$ . Similar arguments can be made with respect to the  $(-x)$ -axis.

### 3.1 Outline of Algorithm

For simplicity, we work in a coordinate frame attached to  $P$ ; the origin is  $cm$  and the  $x$ -axis is parallel to the railing. In this frame, the trap moves in the  $(-x)$ -direction; let  $T(t)$  be the trap at time  $t$ .

For each vertex  $v \in V(t)$ , let  $\beta_v(t) \in [0, 2\pi)$  be its angular position about the origin ( $cm$ ), and let  $L$  and  $U$  be the lower and upper envelopes, respectively, of  $\mathcal{B} = \{\beta_v(t) \mid v \in V(t)\}$  (Figure 4). Then  $P(t)$  drops if  $U(t) - L(t) < \pi$ .

Let  $t_1 < \dots < t_k$  be the set of all breakpoints of  $L$  and  $U$ , set  $t_0 = -\infty$ ,  $t_{k+1} = +\infty$ , and let  $\nu_i$  and  $\chi_i$  be the vertices of  $V$  that realize the lower and upper envelopes, respectively, in the interval  $[t_i, t_{i+1})$ . In order to decide whether  $T$  drops  $P$ , we need to find one value of  $t$  so that  $P(t)$  drops. Our decision algorithm proceeds as follows:

First, compute  $t_i$ ,  $\nu_i$ , and  $\chi_i$ , and check whether  $\beta_{\chi_i}(t_i) - \beta_{\nu_i}(t_i) < \pi$  for any  $t_i$ . If not, check

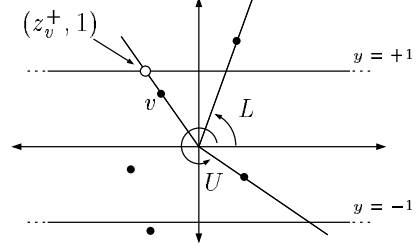


Figure 4:  $L$ ,  $U$ , and  $z_v^+$

whether  $\beta_{\chi_i}(t) - \beta_{\nu_i}(t) = \pi$  for any time  $t \in (t_i, t_{i+1})$ . If necessary, repeat with respect to the  $(-x)$ -axis.  $P$  is supported if no such time is found.

If  $T$  drops  $P$ , then it is also useful to know *all* times when this happens. Let  $s_1 < \dots < s_j$  be the sequence of times which are either breakpoints of  $L$  or  $U$ , or (proper) intersections of  $L + \pi$  and  $U$ . It can be shown that if  $\beta_{\chi_i}(t) - \beta_{\nu_i}(t) \not\equiv \pi$  then there are at most two solutions, so  $j = O(k)$ . To determine  $I = \{t \mid cm \notin CH(S(t))\}$ : Initialize  $I = \emptyset$ , and compute  $s_i$ ,  $\nu_i$  and  $\chi_i$ . Scan the points and intervals in sequence, adding  $\{s_i\}$  to  $I$  if  $\beta_{\chi_i}(s_i) - \beta_{\nu_i}(s_i) < \pi$ , and adding  $(s_i, s_{i+1})$  to  $I$  if  $\beta_{\chi_i}(\frac{s_i+s_{i+1}}{2}) - \beta_{\nu_i}(\frac{s_i+s_{i+1}}{2}) < \pi$ . We then repeat this with respect to the  $(-x)$ -axis, to obtain a list of all times when  $P$  drops.

### 3.2 Implementation Details

Computing  $\mathcal{B}$  explicitly involves inverse trigonometric functions, so we use an alternative parametrization, which is algebraic.

First, partition  $V(t)$  into two sets:

$$\begin{aligned} V_+(t) &= \{v \in V(t) \mid 0 \leq \beta_v(t) < \pi\}, \\ V_-(t) &= \{v \in V(t) \mid \pi \leq \beta_v(t) < 2\pi\}. \end{aligned}$$

If either  $V_+(t)$  or  $V_-(t)$  is empty,  $P(t)$  drops.

For each  $v \in V_+(t)$ , project  $v$  through the origin to the line  $y = +1$  and let  $z_v^+(t)$  be the  $x$ -coordinate of this point. That is,  $z_v^+(t) = \frac{x(v(t))}{y(v(t))} = \cot(\beta_v(t))$ ; see Figure 4. Define  $R_+(t)$  to be the upper envelope of  $\{z_v^+(t) \mid v \in V_+(t)\}$ ; then  $R_+(t) = \cot(L(t))$  if  $0 \leq L(t) < \pi$ ; otherwise,  $R_+(t) = -\infty$ .

Similarly, project  $v \in V_-(t)$  to the line  $y = -1$ , and let  $z_v^-(t) = \frac{x(v(t))}{y(v(t))} = \cot(\beta_v(t))$  (the *negative* of the  $x$ -coordinate of the projected point). Let  $R_-(t)$  again correspond to the rightmost of these projected points; that is, the lower envelope of  $\{z_v^-(t) \mid v \in V_-(t)\}$ . Then  $R_-(t) = \cot(U(t))$  if  $\pi \leq U(t) < 2\pi$  and  $R_-(t) = +\infty$  if  $V_-(t) = \emptyset$ .

It is easily seen that for any  $t$ ,  $U(t) - L(t) < \pi$  if and only if  $R_+(t) - R_-(t) < 0$ . Therefore, if we

replace  $\{s_i\}$  in the algorithms of Section 3.1 with the breakpoints and proper intersections of  $R_+$  and  $R_-$ , and test instead for  $R_+(t) - R_-(t) < 0$ , we will correctly report if and when  $P(t)$  drops.

We have an arrangement of  $O(nm)$  arcs in  $\mathbb{R}^2$  which are algebraic in  $t$  and  $\cot\beta$ . Since any pair of these arcs intersects at most twice, it can be shown [8, Thm 6.5] that the breakpoints of the upper and lower envelopes can be computed in  $O(nm\alpha(nm)\log(nm))$  time, where  $\alpha$  is the inverse of the Ackermann function. We therefore conclude the following

**Theorem 3.1** *Let  $P$  be an oriented polygonal part and  $T$  a polygonal trap with  $n$  and  $m$  vertices, respectively. We can determine whether  $T$  drops  $P$  in  $O(nm\alpha(nm)\log(nm))$  time. Moreover, the same bound holds for finding  $\{t \mid T \text{ drops } P(t)\}$ .*

**Remark 3.2** (i) We can reduce the running time in some cases. If  $P$  and  $T$  can be decomposed into  $k_p$  and  $k_T$   $y$ -monotone pieces, respectively, it can be shown that there are only  $N = O(k_T n + k_P m)$  curve segments in  $\{z_v^\pm\}$ .

(ii) While an edge may define many support vertices at a given time, only the two outer vertices can lie on  $CH(S)$ . By maintaining only these vertices, we can perform the computation using  $O(n+m)$  space.

(iii) This result can be extended to parts and traps with curved edges; specifically, to splinesons, whose boundaries consist of a finite number of algebraic arcs. The modified algorithm runs in  $O(\lambda_s(nm)\log(nm))$  time, where  $\lambda_s(nm)$  is the maximum length of the Davenport-Schinzel sequence of order  $s$  for  $nm$  symbols; see [8]. Both  $s$  and the constant of proportionality depend on the degrees of the polynomials which define the edges.

## 4 Minimal Trap Design

In this section we develop an algorithm for designing a feeder. More precisely: given an oriented polygon  $P_0$ , together with a collection of oriented polygons  $\mathcal{P} = \{P_1, \dots, P_k\}$ , we design a family of feeders which support  $P_0$  and drop each polygon in  $\mathcal{P}$ . In practice, these may be the stable orientations of a single polygon and  $P_0$  the one we want to feed.

We will compute a collection  $\mathcal{T} = \{T_i\}_{i=1}^k$  of traps, where  $T_i$  drops  $P_i$  and supports  $P_0$ . We then place the traps of  $\mathcal{T}$  on the track with no overlap to construct our feeder. (If we place two traps on the track so that they overlap, it is possible that the resulting trap will drop  $P_0$ .)

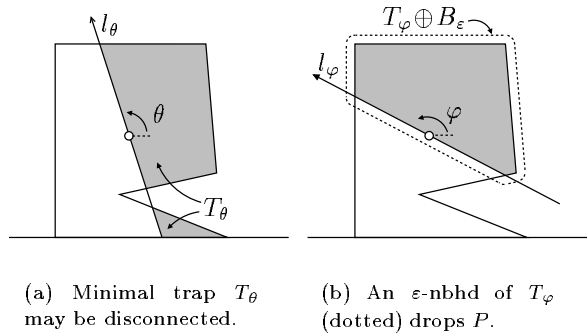


Figure 5: Part  $P$  and two minimal traps (shaded).

We introduce the notion of *minimal traps* in Section 4.1, and Section 4.2 outlines an algorithm to select a subset of minimal traps to use in our feeders. Finally, Section 4.3 addresses whether a part *really* falls and modifies our traps to ensure this.

### 4.1 Minimal Traps

Let  $P$  be a simple polygon in one of its stable orientations, and work in a frame where traps are stationary and  $P$  moves in the  $(+x)$ -direction. For each  $\theta \in \mathbb{S}^1$ , let  $l_\theta$  be the oriented line in the  $\theta$ -direction that passes through  $cm(0)$ . Let  $h_\theta$  be the closed half-plane to the right of  $l_\theta$  and define  $T_\theta = P(0) \cap h_\theta$  (Figure 5). Note that  $l_\theta$  and  $l_{\theta+\pi}$  are different; in fact,  $T_\theta \cup T_{\theta+\pi} = P(0)$ . As above, we remove most of the boundary, so that  $T_\theta = \text{int}(T_\theta) \cup \text{int}(cl(T_\theta) \cap R)$ . The traps  $T_\theta$  are called the *minimal traps* for  $P$ ; they are minimal in the sense that any trap that drops  $P$  must contain one of the  $T_\theta$ .

Unfortunately,  $T_\theta$  does not drop  $P$  at  $t=0$ ; however, any  $\epsilon$ -neighborhood of  $T_\theta$  does (Figure 5(b)). That is,  $(T_\theta \oplus B_\epsilon) \cap \text{Track}$  drops  $P(0)$  for any  $\epsilon > 0$  and any  $\theta$ , where  $\oplus$  is the Minkowski sum, and  $B_\epsilon$  is the disk of radius  $\epsilon$  centered at 0.

The issue here is the presence of the boundary of  $T$ ;  $\partial P$  causes similar problems. A pose which is supported at  $t$  but would drop if either  $\partial T$  were removed from the track or  $\partial P$  were removed from the part is not desirable, as slight fluctuations in the shape of the part or trap make it impossible to predict whether  $P$  drops. We say  $P$  is *semi-supported* at  $t$  in this case, and reserve “supported” for parts that are neither dropped nor semi-supported.

It is not difficult to show that  $T$  supports  $P$  if and only if  $cm(t) \in \text{int}(CH(V(t) - V_T(t)))$ , where  $V_T(t) \subset V(t)$  is the set of isolated vertices of  $S(t)$ . Furthermore, if  $T$  supports  $P$ , then there exists an

$\varepsilon_0 > 0$  such that  $T \oplus B_\varepsilon$  supports  $P$  for all  $0 < \varepsilon < \varepsilon_0$ ; that is, we can make the trap  $\varepsilon$  larger without dropping  $P$ .

To design a feeder, we choose one minimal trap for each  $P_i$  and expand it by an appropriate  $\varepsilon_i$ . By avoiding the minimal traps that only semi-support  $P_0$ , and choosing  $\varepsilon_i$  sufficiently small, we can be sure that this expanded trap supports  $P_0$  as well.

## 4.2 Choosing a Minimal Trap

Our goal in this section is to determine which of the minimal traps for  $P_1$  support  $P_0$ .

We use the previous algorithm to test whether  $T_\theta$  drops  $P_0$ , where  $T_\theta$  is a minimal trap for  $P_1$ . If we denote by  $V(t, \theta)$  the support vertices of  $P_0(t)$  with respect to  $T_\theta$ , then we want to determine

$$\Theta = \{\theta \mid cm_0(t) \in \text{int}(CH(V(t, \theta) - V_I(t, \theta))) \forall t\},$$

where  $cm_0$  is the center of mass of  $P_0$  and  $V_I$  is the set of isolated vertices of  $S(t, \theta) = P(t) - T_\theta$ .

Define  $z_v^\pm(t, \theta)$  as before. Let  $R_+(t, \theta)$  be the upper envelope of the collection of surface patches  $\{z_v^+(t, \theta) \mid v(t) \in V_+(t, \theta) - V_I(t, \theta)\}$  and similarly for  $R_-(t, \theta)$ . By removing the isolated vertices, we leave open the corresponding boundary edges of the  $z_v^\pm(t, \theta)$  surfaces.

Define  $\mathcal{D}_R = \{(t, \theta) \mid R_+(t, \theta) - R_-(t, \theta) \leq 0\}$ . These correspond to poses which are either dropped or semi-supported, since  $cm \in \partial CH(V - V_I)$  when  $R_+ - R_- = 0$ . Similarly define  $L_\pm$  and  $\mathcal{D}_L$  with respect to the  $(-x)$ -axis. Then  $\theta \in \Theta$  if and only if the line  $\mathbb{R} \times \{\theta\}$  does not intersect

$$\mathcal{D} = \mathcal{D}_R \cup \mathcal{D}_L = \{(t, \theta) \mid cm \notin \text{int}(CH(V - V_I))\}.$$

Each of these sets can be determined by computing an appropriate envelope:  $\mathcal{D}_R$  is the projection to the  $t$ - $\theta$  plane of the portion of the lower envelope of  $\{R_+, R_-\}$  that is realized by  $R_+$ , and similarly for  $\mathcal{D}_L$ .  $\Theta$  is the set of all  $\theta$  for which the left envelope of the boundary curves of  $\mathcal{D}$  is  $+\infty$ .

Assuming  $P_0$  and  $P_1$  both have  $O(n)$  vertices, we have an arrangement of  $O(n^2)$  surface patches which are algebraic in  $t$ ,  $\cot \beta$  and  $\tan \theta$ . It can be shown [8, Ch. 7] that the upper and lower envelopes can be computed in  $O(n^{4+\varepsilon})$  time. In fact, since for any fixed  $(t, \theta)$  pair we need only  $O(n)$  supporting vertices, we can improve this to  $O(n^{3+\varepsilon})$ . Thus the total time to design our family of feeders is  $O(k n^{3+\varepsilon})$ , where  $k = |\mathcal{P}|$ .

**Theorem 4.1** *Let  $P_0$  be an  $n$ -sided polygonal part in one of its stable orientations and let  $\mathcal{P} = \{P_1, \dots, P_k\}$  be a collection of  $k$  polygons, each*

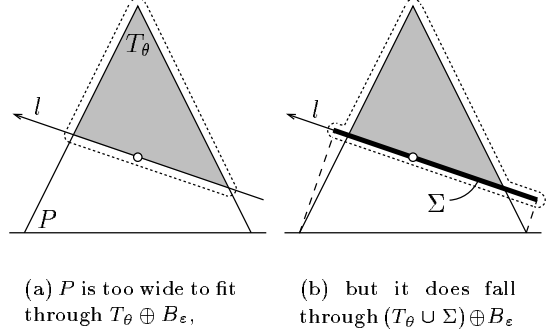


Figure 6: Adding  $\Sigma$ , the bold part of  $l$ , allows  $P$  to fall.

*of which has  $O(n)$  vertices. We can compute in  $O(k n^{3+\varepsilon})$  time a family of feeders that drops  $\mathcal{P}$  and supports  $P_0$ , if such a feeder exists.*

## 4.3 On Falling

We mentioned early on that “ $T$  drops  $P$ ” does not guarantee that the part actually falls through the trap. In this section we describe how to modify a trap so that a part which it drops really falls. If the polygons are all convex, we are then able to use these modified minimal traps to design a complete family of feeders as before. In the non-convex case the solution is no longer complete, but is nonetheless useful as a heuristic.

Let  $P$  be any oriented polygon and  $T$  a trap that drops it. We make the following assumptions about how  $P$  falls into  $T$ : The first instant it becomes unsupported, say at  $t_0$ ,  $cm$  will follow the shortest possible downward path. This causes the part to move as follows: Let  $v$  be the nearest point on  $\partial CH(S(t_0))$  to  $cm(t_0)$ . As  $P$  falls, it pivots about the line through  $v$  orthogonal to the segment  $\overline{vcm}$ . (If  $cm \in \partial CH(S)$  at  $t_0$ , this line is simply the edge containing  $cm$ .) We call this line the *drop line* for  $P$  with respect to  $T$ .

After it begins to tip, we assume that the part rotates about the drop line until it is vertical, then slides straight down if it can. There are two situations which can keep it from doing so. First, the part may be too wide. Either it is too wide on both sides, and the part cannot slide through (Figure 6(a)); or only one side is too wide (Figure 5(b)), and the part may still slide through, but only after rotating in its vertical plane. Second, the trap may have more than one connected component, and get stuck straddling a portion of the track between components (Figure 5(a)).

Let  $T_\theta$  be a minimal trap for  $P$  and fix any  $\varepsilon > 0$ ;

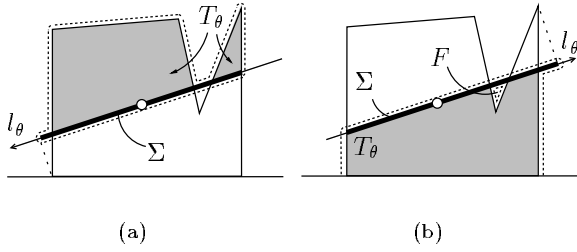


Figure 7: The addition of  $\Sigma$  may (a) join disconnected pieces of  $T_\theta$  or (b) create a hole ( $F$ ) in the trap.

then  $T_\theta \oplus B_\varepsilon$  drops  $P$ . Let  $l$  be the drop line for  $P$  with respect to  $T_\theta \oplus B_\varepsilon$ , let  $S_\varepsilon = P - (T_\theta \oplus B_\varepsilon)$ , and let  $\Sigma$  be the convex hull of the orthogonal projection of  $S_\varepsilon$  onto  $l$  (Figures 6(b), 7).

Using  $(T_\theta \cup \Sigma) \oplus B_\varepsilon$  for the trap almost works, except for two problems. First, we cannot extend the trap below  $R$  and need another solution when  $\Sigma$  intersects the railing. Second, if  $T_\theta \cup \Sigma$  is not simply connected (Figure 7(b)), it is not a realistic trap as it implies a free-floating portion of the track.

If  $\Sigma \cap R \neq \emptyset$ , the part cannot slide straight down once it has tipped upright. However, by rotating in the vertical plane which contains the drop line, the part can slide through, provided that there is a sufficiently long open strip along the drop line. We therefore define  $\Sigma$  in this case to be the segment of the drop line starting at  $R$  whose length is the diameter of the part. (While this is likely longer than necessary, it is certainly always long enough.)

If  $T_\theta \cup \Sigma$  is not simply connected, let  $F$  be the bounded components of  $\text{Track} - (T_\theta \cup \Sigma)$ . In this situation, we must add  $F$  to our trap. We therefore define our modified minimal traps:

$$\hat{T}_\theta(\varepsilon) = (T_\theta \cup \Sigma \cup F) \oplus B_\varepsilon.$$

Return now to the the feeder design problem, and let  $\hat{T}_\theta$  be the modified minimal traps for  $P_1$ . We need to determine  $\hat{\Theta} = \{\theta \mid \hat{T}_\theta \text{ supports } P_0\}$ .

First, note that the addition of  $\Sigma$  does not affect whether  $P_0$  is supported. Moreover, if  $P_1$  is convex, then  $F = \emptyset$  and  $\hat{\Theta} = \Theta$ . Thus we can use the algorithm of Section 4.2 to find a feeder built from these modified minimal traps, as long some feeder exists.

Even in the non-convex case, the algorithm can be modified to compute  $\hat{\Theta}$  in  $O(k n^{3+\varepsilon})$  time, where  $k$  is the number of polygons to be dropped; we omit the details. However, if  $P_1$  is not convex,  $F$  may be non-empty. Since the addition of  $F$  to the trap may cause  $P_0$  to drop,  $\hat{\Theta}$  may be strictly smaller

than  $\Theta$ . In fact, there is a chance that even though the previous algorithm generates feeders, it is impossible to modify them and still support  $P_0$ : A modified minimal trap  $\hat{T}_\theta$  for  $P_1$  may drop  $P_2$ , but to modify  $\hat{T}_\theta$  so that  $P_2$  falls we add an  $F$ , thus altering  $\hat{\Theta}$  or even the drop line for  $P_1$ , etc. While the solution is no longer complete, avoiding angles in  $\hat{\Theta}_i$  for which the other  $P_j$  drop still provides a useful heuristic for designing more realistic feeders for non-convex parts.

## 5 Conclusion

In this paper, we furthered the geometric approach to the trap-based feeder-design problem introduced in [3]. We presented an improved algorithm for analyzing a polygonal trap with respect to a part, introduced the notion of minimal trap, and developed an algorithm for designing a complete family of feeders from them.

Open problems include classifying all parts for which no feeder exists, finding an optimal way to overlap the minimal traps, addressing tolerance issues, designing more efficient algorithms, and extending the algorithm to three dimensional parts.

## References

- [1] J. Basch, L. J. Guibas, and J. Hershberger. Data structures for mobile data. *Proc. 8th ACM-SIAM Sympos. Discrete Algorithms*, pages 747–756, 1997.
- [2] R.-P. Berretty. *Geometric Design of Part Feeders*. PhD thesis, Utrecht University, Utrecht, The Netherlands, 2000.
- [3] R.-P. Berretty, K. Goldberg, M. H. Overmars, and A. F. van der Stappen. Geometric algorithms for trap design. *Symp. Comp. Geom.*, pages 95–104, 1999.
- [4] G. Boothroyd, C. Poli, and L. Murch. *Automatic Assembly*. Marcel, Dekker, Inc., New York, 1982.
- [5] A. Christiansen, V. Chandry, and M.M. Barash. Automatic design of parts feeders using a genetic algorithm. In *Proc. IEEE ICRA*, pages 846–851, 1996.
- [6] K. Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10(2):201–225, August 1993.
- [7] L. Lim, B. Ngoi, S. Lee, S. Lye, and P. Tan. A computer-aided framework for the selection and sequencing of obtaining devices for the vibratory bowl feeder. *Intl. J. Production Research*, 32, 1994.
- [8] M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, 1995.