TRIANGULATING A NONCONVEX POLYTOPE

Bernard Chazelle
Leonidas Palios

CS-TR-227-89

August 1989

Triangulating a Nonconvex Polytope

BERNARD CHAZELLE AND LEONIDAS PALIOS *
Department of Computer Science
Princeton University
Princeton, NJ 08544

Abstract: This paper is concerned with the problem of partitioning a three-dimensional polytope into a small number of elementary convex parts. The need for such decompositions arises in tool design, computer-aided manufacturing, finite-element methods, and robotics. Our main result is an algorithm for decomposing a polytope with n vertices and r reflex edges into $O(n+r^2)$ tetrahedra. This bound is asymptotically tight in the worst case. The algorithm is simple and practical. Its running time is $O\left((n+r^2)\log r\right)$.

1. Introduction

This work is concerned with the problem of partitioning a polytope in $\Re^3$ into a small number of elementary convex parts. The general problem of decomposing an object into simpler components has been the focus of much attention in recent years. In two dimensions, computer graphics and pattern recognition have been the main source of motivation for this work. Beginning with the papers of Feng and Pavlidis [15] and Schachter [24], the problem of rewriting a simple polygon as a collection of simple parts has been exhaustively researched, cf. O'Rourke's book [21] and the survey article (Chazelle [8]). In higher dimensions, however, results have been few and far between. It is known from (Chazelle [7]) that a polytope of n vertices can always be partitioned into $O(n^2)$

* This research was supported in part by the National Science Foundation under Grant CCR–8700917.

convex pieces, provided that Steiner points are allowed in the decomposition, and that this bound is tight in the worst case. If no Steiner points are allowed, then Ruppert and Seidel [23] have recently proven that the problem of deciding whether a given polytope is decomposable into tetrahedra (whose vertices have to be vertices of the polytope) is NP-complete, even if restricted to the class of star-shaped polytopes. The algorithm to carry out the partitioning described in (Chazelle [7]) is based on the exact RAM model of computation. The issue of robustness while dealing with finite precision arithmetic is considered in (Bajaj and Dey [2]), where a numerically robust algorithm to compute a convex decomposition of non-convex polytopes of arbitrary genus is presented.

On a related problem, Aronov and Sharir [1] have shown that the cells of an arrangement of n triangles in 3-space can be partitioned into a total of $O(n^2\alpha(n) + h)$ tetrahedra, where h is the number of faces in the arrangement, and $\alpha(n)$ is the inverse Ackermann function. For fixed arbitrary dimension d, Edelsbrunner et al. [14] have given an optimal algorithm for computing the partition of d-space induced by a collection of hyperplanes. The stratification of real-algebraic varieties and related issues are discussed in [6,10,11,22,25,27].

The specific problem of partitioning a three-dimensional polytope into simple parts arises in mesh-generation for finite-element methods, computer-aided design and manufacturing, automated assembly systems and robotics (Baker [3], Smith [26]). The problem comes under various guises, depending on the desired shape of partitioning elements: convex, simplicial, star-shaped, monotone, rectangular, isothetic, etc. In general, the quest for minimal partitions seems destined to be frustrating. For example, finding minimum convex decompositions is NP-hard (Lingas [18]). In practice, however, good approximation algorithms may be just as attractive, especially, if the decomposition is fast, robust, and free of pathological features. Indeed, a minimum partition can be sometimes so contrived that a finer, yet more regular, decomposition is preferable.

How difficult is it to triangulate a polytope (that is, subdivide it into a collection of tetrahedra)? In practice, a "good" triangulation algorithm should not only guarantee $O(n^2)$ pieces in the worst case, but it should also make the size of the triangulation dependent on both n, the size of the polytope, and r, the number of reflex edges. The polytopes arising in standard applications areas tend to be quasi-convex, and this fact should be used to one's advantage. For example, a triangulation of quadratic size would be disastrous if, say, the polytope is convex. When both n and r are taken into account, the lower bound on the triangulation size becomes $\Omega(n+r^2)$ (as is easily derived from [7]). By this criterion, the algorithm described in this paper is optimal: A polytope of n vertices and r reflex edges is triangulated into $O(n+r^2)$ pieces. The running time is $O((n+r^2)\log r)$. The algorithm is very simple and we believe that it will be practical. Plans are under way to implement it and test it on actual problems arising in the use of finite-element methods in aerospace engineering.

The triangulation algorithm consists of two parts. In a *pop-out phase* we identify vertices of small degree that are not *hindered* by other vertices and remove them one by one, much like we would pull out a ski hat off someone's head. This pruning operation reduces the size of the polytope to $O(r)$. Next, we enter the *fence-off phase*, which involves erecting vertical fences from each edge of the polytope. We use section 2 to set our notation and move a number of technicalities out of the way. Section 3 describes the triangulation algorithm proper.

2. Cups, Crowns, Domes, and Other Widgets

We begin by recalling some standard terminology and introducing some of our own. Let P be a polytope in $\Re^3$. We assume that P is simple, meaning that it is a piecewise-linear 3-manifold with boundary, which is homeomorphic to a closed 3-ball. We also assume that its boundary ∂P is facially structured as a two-dimensional cell complex. Its elements are relatively open sets which are called ***vertices***, ***edges***, or ***facets***, if their affine closures have dimension 0, 1, or 2, respectively. Simplicity rules out handles, dangling or abutting edges (figure 1), but it allows facets to have holes. An edge e of P is said to be ***reflex*** if the (interior) dihedral angle formed by its two incident facets exceeds π. By extension, we say that a vertex is ***reflex*** if it is incident upon at least one reflex edge, and that it is ***flat*** if all its incident facets lie in at most two distinct planes. Finally, a vertex is ***pointed*** if it is neither flat nor reflex (figure 2). It is easy to see that a pointed vertex cannot be incident upon two collinear edges, although it can be incident upon two coplanar (adjacent) facets of P. Next, we define the ***cone*** of a pointed vertex v as the unbounded convex polyhedron spanned by the edges incident upon v. More precisely, $cone(v)$ is the locus of points $v + \sum_{1 \le i \le k} \alpha_i (w_i - v)$, where $w_1, \ldots, w_k$ are the vertices of P adjacent to v and the α_i's are arbitrary nonnegative reals. We are now ready to introduce the key notion of a ***cup***. The cone of a pointed vertex v contains a number of vertices of P distinct from v. Some lie on the boundary of the cone; others may lie strictly inside. The cup of v is a portion of the cone which contains v but steers clear of the other vertices. Let K^+ and K be the convex hulls of all the vertices of P lying in $cone(v)$ and $cone(v) \setminus \{v\}$, respectively. We define $cup(v)$ as the simple polytope formed by the closure of $K^+ \setminus K$. If a vertex of P intersects the cup outside the cone's boundary but is *not* a vertex of the cup by virtue of the definition given above, then we make it into one and call it ***stranded***. As we shall see below, stranded vertices can only lie on the boundary of the cup and thus sit on facets with or without cup edges incident upon them (figure 3).

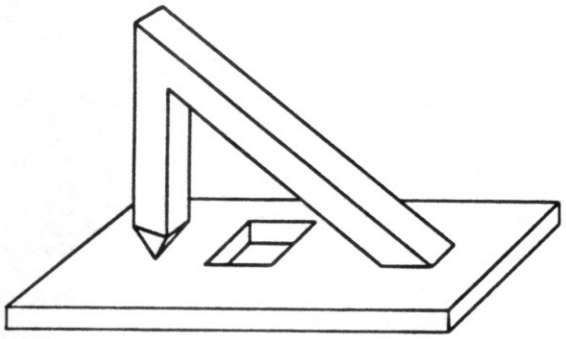
Figure 1: A nonsimple polytope

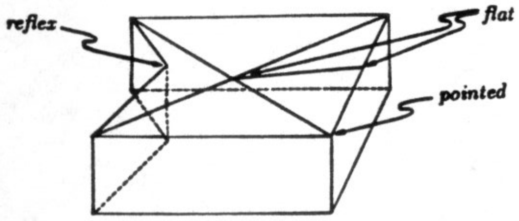

Figure 2: The different types of vertices

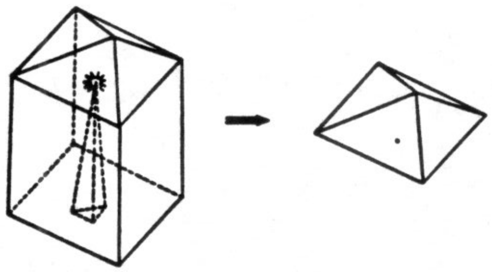
Figure 3: A cup with a stranded vertex

A number of simple properties follow readily from the definition. A cup is the closure of the difference between the convex hull of a finite point-set $A \sqcup \{v\}$ and the convex hull of A. Since v does not belong to A, its cup is a simple star-shaped polytope whose kernel contains v (figure 4). Its boundary contains a number of polygons incident upon v which are glued to the convex hull of A. The glueing border can be centrally projected onto a plane so as to appear as the boundary of a convex polygon. The border in question is a closed simple polygonal curve, called the *crown* of v. The crown acts as a Jordan curve on the boundary of the cup, which it separates into one piece on the boundary of the cone and a convex polyhedral patch, which we call the *dome* of v. Edges and vertices of the dome that are not in the crown are called *internal*. Obviously, the internal edges of the dome are the only edges of the cup which are reflex (with respect to the cup). To conclude this string of definitions, we refer to the pointed vertex v as the *apex* of $cup(v)$.

We now investigate the relationship between P and the cup of v. All cup vertices are vertices of P though, obviously, the same cannot be said of cup edges. A more interesting observation is that the cup lies inside P. This follows from the fact, to be proven below, that the facets of the cup that are not in the dome lie in ∂P. Thus, it is impossible for a facet or an edge of P to intersect the interior of the cup, unless a vertex of P does. But that, of course, is ruled out by the very definition of a cup. This establishes our claim and also shows that stranded vertices can only be internal vertices of the dome. Let us now prove the premise of this reasoning, which is that a facet of the cup that is not in the dome lies on the boundary ∂P. It suffices to show that the crown lies entirely in ∂P. Let $g_1, \ldots, g_\ell$ be the facets of P incident upon v and let us (mentally) merge any pair of coplanar facets. This produces superfacets $f_1, \ldots, f_k$ (given in either circular order around v), such that the dihedral angle between two adjacent f_i's is strictly less than π. Now, for each $i = 1, \ldots, k$, let K_i^+ (resp. K_i) be the two-dimensional convex hull of the vertices of P lying in f_i (resp. $f_i \setminus \{v\}$). The closure of $K_i^+ \setminus K_i$ is the polygon formed by intersecting the cup of v with the plane supporting f_i: it is the two-dimensional equivalent of a cup (figure 5). Its boundary consists

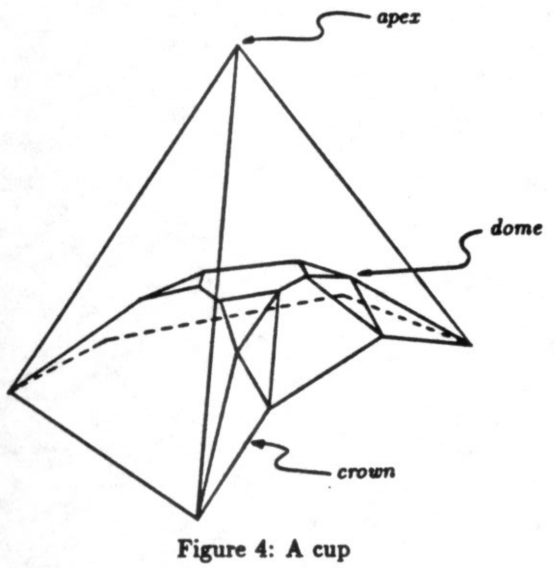

Figure 4: A cup

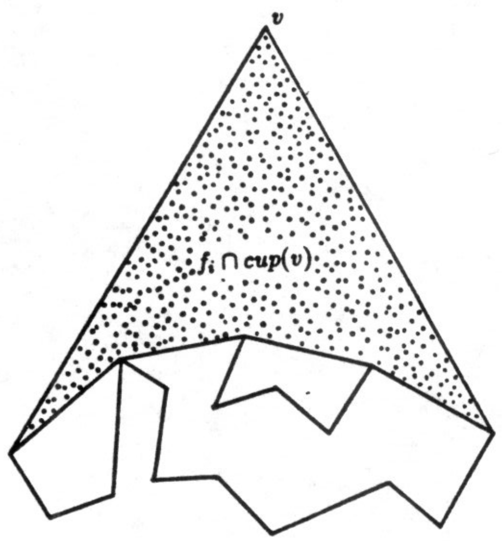

Figure 5: A facet of a cup

of a two-edge convex chain followed by a concave chain (possibly reduced to a single edge). By a convex (resp. concave) chain we mean a piece of a polygon's boundary which always turns strictly right (resp. left) when traversed clockwise. The construction works as desired because v is pointed, and therefore exhibits an angle less than π in f_i. The crown of v is the closed curve obtained by concatenating the concave chains in sequence. This proves our claim that the crown lies in ∂P. Additionally, no vertices but the endpoints of such concave chains can be pointed vertices of P. Therefore, since all edges of the cup adjacent to the apex are also edges of P, a pointed vertex can be on the crown of another pointed vertex only if they are adjacent in P. Note however, that the converse is not always true. In particular, if two pointed vertices are connected by an edge which is incident upon two coplanar facets, none of them lies on the crown of the other.

Let us summarize the various types of faces which a cup may have. The following statements are to be understood with respect to the cup and ***not*** P. The edges incident to the apex as well as the edges of the crown are nonreflex. Actually, none of them can be incident to two coplanar facets. The reason is that the convex hull operation merges coplanar facets. As a result, although each facet of the cup incident upon v lies in ∂P, it does not necessarily lie within any given facet of P. Returning to our classification, we should note that the internal edges of the dome are all reflex.

We close this section with a few technical lemmas which hold the key to understanding the whys and wherefores of the pop-out phase. That phase involves identifying pointed vertices of small degree whose domes are unhindered. We say that a dome is ***hindered*** if it contains (i) an internal vertex, or (ii) an internal edge that is also an edge of P. As usual, the ***degree*** of a vertex of P refers to the number of edges incident upon it. The idea is to pull out such a desirable vertex by removing the boundary of its cup and replacing it by its dome. This shelling step decreases the vertex count by one without increasing the number of reflex edges.

In the following, we assume that P is a simple polytope with n vertices and m edges, exactly r of which are reflex. We shall also assume that P does not have any flat vertices.

Lemma 2.1. *Let v and v' be two distinct nonadjacent pointed vertices of a simple polytope. No point can be an internal vertex for the domes of both v and v'. Similarly, no line segment can be an internal edge of both domes.*

Proof: Let z be a vertex internal to the domes of v and v'. Let us first assume that the intersection Q of the interiors of the cups of v and v' is nonempty. The closure of Q must have at least one vertex outside the dome of v, otherwise it would have empty interior. The only such vertex can be the apex v, however, since the interior of a cup is free of vertices. Thus, v lies in the cup of v'. Since it can neither coincide with v' nor be an internal vertex of the dome of v', the vertex v must lie in the crown of v'. But this was mentioned earlier as an impossibility, since v and v' are nonadjacent.

So, we can now assume that the intersection of the interiors of the two cups is empty. Since z is internal to the dome of v, there exists a small open half-ball centered at z that lies entirely within the cup of v. A similar statement holds for v' as well, and since the two half-balls are nonintersecting, the domes of both v and v', locally around z, have to lie on the plane separating the two half-balls, which contradicts the simplicity of P.

This proves the first part of the lemma. The second part is a trivial corollary: simply introduce an artificial vertex at the midpoint of the internal edge. ∎

Lemma 2.2. *Any reflex vertex which is internal to a dome has at least three reflex edges incident upon it.*

Proof: Let w be an internal vertex of the dome of some pointed vertex. There exists a small open half-ball centered at w that lies entirely inside P. Therefore, w is a vertex of the convex hull of the point-set consisting of w and of the endpoints of the edges incident upon it. Note that the convex hull edges incident upon w are also edges of the polytope, and are in fact reflex. From the simplicity of P, we can now conclude that at least three reflex edges of the polytope are incident upon w. ∎

Lemma 2.3. *A reflex vertex of P can contribute internal vertices to at most three distinct domes. Similarly, a reflex edge of P can contribute internal edges to at most three domes.*

Proof: Figure 6 shows that these bounds are tight. Now, suppose, for contradiction, that a reflex vertex p is internal to the domes of four pointed vertices u, v, w, z of P. It follows from Lemma 2.1 that all four apexes must be adjacent to each other, thus forming a tetrahedron T whose six edges all lie in ∂P. This tetrahedron cannot have empty interior, otherwise one of the apexes would be either reflex or flat. Note also that the cup of a pointed vertex s contains any vertex t adjacent to s such that the (interior) dihedral angle around st is strictly less than π. Therefore, the crown of each of u, v, w, z contains the other three apexes as vertices. Furthermore, since p is an internal vertex of all four domes, and the edges of T are nonreflex, p must lie in T. However, it cannot lie on any of T's edges, otherwise there would be two nonadjacent apexes.

Since p is internal to the domes of all four apexes, there exists a small ball centered at p that lies entirely in each of their cones. Because P is simple, the ball intersects the complement of P, and so, in particular contains a point q outside of P that avoids each of the six planes defined by p and any two of the four apexes. It follows that p must lie in the relative interior of one of the four tetrahedra defined by q and any three of the apexes. All four vertices of that tetrahedron, however, lie in the cone of the fourth apex, which prevents p from being a vertex of its dome, and gives us a contradiction.

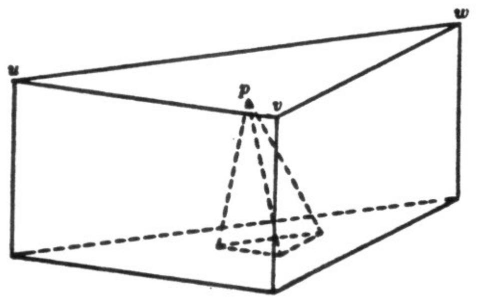

Figure 6: A reflex vertex contributing internal vertices to three domes

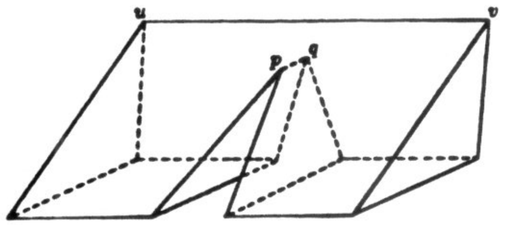

Figure 7: An internal edge with its endpoints on the crowns of two domes

As in Lemma 2.1, to prove the second part, we introduce an artificial reflex vertex at the midpoint of the edge, and make use of the previous result. ∎

Lemma 2.4. *Given an edge pq of P, there are at most two pointed vertices v and w, such that pq is internal to the domes of both v and w, and p and q lie on the crowns of both domes.*

Proof: Suppose that pq is internal to the domes of v, w, z, and that p and q lie in all three crowns. Then, all six line segments pv, qv, pw, qw, pz, qz lie in ∂P, and all three triangles pvq, pwq, pzq lie entirely in P. Since v, w, z are pointed vertices, the boundary of each of these triangles cannot contain more than one of the apexes, and thus a total ordering of them around pq can be established. Furthermore, since the genus of ∂P is 0, these boundaries act as Jordan curves, partitioning ∂P into two polyhedral patches each. Therefore, there is one apex, say v, such that each of the other two apexes w, z belongs to each of the two polyhedral patches delimited by pv, qv, and pq. It follows from Lemma 2.1 that v, w, z must be adjacent to each other, and so the edge wz must cross one of the segments pv, qv, or pq. In the first two cases, p and q, respectively, would be excluded from the cup of v. In the last case the two facets incident upon pq would be coplanar, which contradicts the fact that pq is a reflex edge. Note that the statement of the lemma is tight, as shown in figure 7. ∎

Lemma 2.5. *The polytope P contains at most $2r$ pointed vertices whose domes are hindered.*

Proof: We partition the reflex edges of P into three classes:

1: those with at least one endpoint being an internal vertex of some dome,
2: internal edges of a dome with both endpoints on the crown, and
3: all remaining reflex edges.

Let us prove by contradiction that classes 1 and 2 are disjoint. Assume that there exists a reflex edge e of P, internal to the domes of two pointed vertices u and v, such that one of its endpoints, say p, lies on the crown of u and is internal to the dome of v. Then, there exists a small open half-ball centered at p that lies entirely in the cup of v, and hence in P. Since e is internal to the dome of u, it is not collinear with pu. With pu nonreflex, it follows that the unique plane defined by e and pu intersects the half-ball in a half-disk centered at p. The internal angle between e and pu is strictly less than π however, therefore the half-disk cannot lie entirely in P: a contradiction. We conclude that no vertex can be internal to the dome of a pointed vertex and on the crown of another one.

Let r_i $(i = 1, 2, 3)$ be the cardinality of the i-th class above. According to Lemma 2.3, an edge in class 1 may hinder at most three domes by contributing internal vertices through one given endpoint q (so the total might be as high as 6). Note that if an endpoint of such an edge lies on the crown of some pointed vertex, then, according to the argument above, it cannot be internal to, and thus cannot hinder any dome. But the endpoint q will be incident upon at least two additional reflex edges of P in class 1 (Lemma 2.2). Therefore, the r_1 edges in class 1 can hinder at most $(2 \times 3)/3\ r_1$ domes. Additionally, from Lemma 2.4, each edge in class 2 may hinder at most two domes. The lemma follows readily. ∎

Lemma 2.6. *Let $d \geq 6$ be a fixed integer. If $r < (d-5)n/(2d)$ then P contains at least*

$$\frac{d-5}{d-2}n - \frac{2d}{d-2}r + \frac{12}{d-2}$$

pointed vertices of degree at most d.

Proof: Since a pointed vertex is incident upon nonreflex edges only,

$$\sum_i d_i \leq 2(m-r),$$

where the sum of the degrees d_i extends over all pointed vertices. Furthermore,

$$\sum_i d_i = \sum_{3 \leq d_j \leq d} d_j + \sum_{d_j > d} d_j \geq 3N + (n - n' - N)(d+1),$$

where N is the number of pointed vertices of degree at most equal to d, and n' the number of reflex vertices. The combination of the last two inequalities yields $3N + (n-n'-N)(d+1) \leq 2(m-r)$. Therefore,

$$N \geq \frac{(d+1)(n-n') - 2(m-r)}{d-2}.$$

Since each reflex vertex is incident upon at least one reflex edge, we have $n' \leq 2r$, while $m \leq 3n-6$, from Euler's relation. Substituting these values, we derive

$$N \geq \frac{(d+1)n - (d+1)2r - 2(3n-6) + 2r}{d-2} = \frac{(d-5)n - 2dr + 12}{d-2}. \quad \blacksquare$$

Lemma 2.7. *Let $d \geq 6$ be a fixed integer. If $r = c(d-5)n/(4d-4)$ for some $c < 1$, then P contains at least $(1-c)(d-5)n/(d-2)$ pointed vertices of degree $\leq d$ whose domes are unhindered.*

Proof: From Lemma 2.6, we derive a lower bound on the number of pointed vertices of degree at most d. Among these vertices, at most $2r$ can have their domes hindered (Lemma 2.5). So, in order to guarantee the presence of pointed vertices with unhindered domes, it suffices to have

$$\frac{d-5}{d-2}n - \frac{2d}{d-2}r + \frac{12}{d-2} > 2r.$$

The number of such vertices will be at least

$$\frac{d-5}{d-2}n - \frac{2d}{d-2}r + \frac{12}{d-2} - 2r = \frac{(d-5)n - (4d-4)r + 12}{d-2} \geq \frac{(1-c)(d-5)n + 12}{d-2}. \quad \blacksquare$$

3. The Triangulation Algorithm

Given a simple polytope P with n vertices and r reflex edges, we show how to partition P into $O(n + r^2)$ tetrahedra. The algorithm requires $O(n \log r + r^2 \log r)$ time and $O(n + r^2)$ space. Up to within a constant factor, the number of tetrahedra produced by the algorithm is optimal in the worst case. This follows from a lower bound of $\Omega(m^2)$ on the number of convex parts needed to partition a certain polytope of m vertices, which is a member of an infinite family $\{P_m\}$ (Chazelle [7]). Indeed, we simply add dummy nonreflex edges to P_r until we have a polytope of n vertices with r reflex edges. Although not all realizable pairs (n, r) might be obtained in this way, enough of them are to justify our claim that $\Theta(n+r^2)$ is a tight worst-case bound on the number of tetrahedra needed to triangulate a polytope with n vertices and r reflex edges.

As we alluded to earlier, the triangulation algorithm consists of two phases, figuratively termed *pop-out* and *fence-off*. We shall assume that the polytope P is free of flat vertices and is given to us in *normal* form, meaning that its boundary is triangulated and that all incidences are explicitly listed. For this purpose, we can use any of the standard polyhedral representations given in the literature, e.g., winged-edge (Baumgart [5]), doubly-connected-edge-list (Muller and Preparata [20]), quad-edge (Guibas and Stolfi [16]).

A simple polytope can be normalized in $O(n + r \log r)$ time using, say, Mehlhorn and Hertel's triangulation algorithm [19]. To do so, we triangulate each facet by sweeping a line across its supporting plane, stopping only at vertices exhibiting reflex angles. Since these vertices are incident upon reflex edges, there will be at most $O(r)$ sweep-line stops, each incurring a search and update cost of $O(\log r)$ time. Note that the normalization may increase the number of nonreflex edges, but does not affect n or r.

A. The fence-off phase. Our goal here is to triangulate a simple polytope of n vertices into $O(n^2)$ tetrahedra. This method works well when at least a fixed fraction of the edges are reflex. We begin by partitioning P into cylindrical pieces, and then we triangulate each piece separately. To build the cylindrical partition we attach vertical fences to each edge, reflex and nonreflex, one at a time. Let us say that a point p is *visible* from an edge if it can be connected to it by a vertical segment whose relative interior lies in the interior of P. The set of points visible from e is easily seen to be a monotone polygon: it is called the *fence* of the edge e and is to be attached to it. Our fences are similar to the *walls* used in the *slicing theorem* of (Aronov and Sharir [1]): one difference is that while fences project vertically onto their attaching edges, walls flood all over the free portion of the vertical plane passing through the edge. Three questions arise: (i) How do we erect a fence? (ii) Does fencing ensure convexity? (iii) How many new edges do we create in the process? We shall answer all three questions in that order.

In general, erecting fences will result in cutting off some edges into sub-edges. We shall still treat each edge as one entity, and deal with all its sub-edges in one fell swoop. Note that because of sub-edges, the fence attached to an edge e of P might be itself decomposed into monotone sub-pieces separated by vertical edges. We give a very simple, albeit slightly inefficient, method for computing the fence of e. Let Σ be the set of segments obtained by computing the intersection of ∂P and all previous fences with the vertical plane passing through e. In general, Σ forms disjoint polygonal boundaries augmented with vertical segments created by previous fences. Next, we compute the trapezoidal map induced by the visibility relation among the segments of Σ. This is the planar partition formed by Σ and all the vertical segments that connect endpoints to their visible segments in Σ. Since the number of fences is $O(n)$ and none can contribute more than a single segment to Σ (because of the monotonicity of the visibility polygons), the size of the trapezoidal map is $O(n)$. The collection of regions incident upon e partitions its fence into trapezoids (figure 8). The entire computation for each edge can be carried out in $O(n \log n)$ time.

Once the fence is available, it should be added on to P and the previous fences. Our approach, however, will be not to explicitly store each fence. The fencing operation partitions P into cylindrical pieces, each of which can be defined by (i) specifying a horizontal base polygon, (ii) lifting it vertically into an infinite cylinder, and (iii) clipping the cylinder between two planes (which do not intersect inside the cylinder). In order to extract any of these cylindrical pieces from the polytope, all we need is to compute the shattering that the fences induce to the facets of P, and to be able to locate fast the opposite base of any such piece, given its other base polygon. So, we associate with each facet a list of line segments along which the fences intersect the facet. Then, for each fence, we need only update the lists of the appropriate facets. Additionally, we establish pointers between the non-parallel edges of each trapezoid in each fence, which allow for constant time location of the opposite base of any cylindrical piece. After all fences have been added in this way, the shattering of each facet can be accomplished by sorting the endpoints of the line segments in the facet's list, and sweeping the facet processing the line segments in order.

Note that, at any given time, some fences may have exposed edges "sticking out". The hope is that in the end P will be nicely decomposed into convex polytopes. Unfortunately, this is not always true. Of course, every reflex edge of P is "resolved" in the sense that the angles between its adjacent facets cease to be reflex. The problem is that new reflex edges might be created between two fences (figure 9). Let us examine this phenomenon in some detail. Let e be a vertical edge of a fence. Since the edge e is incident upon at least one vertex of P, and ∂P is triangulated, the edge e cannot be left exposed after the fencing. It is conceivable, however, that e coincides with an edge of another fence which results in a reflex edge. What we can say at this point is that the fences

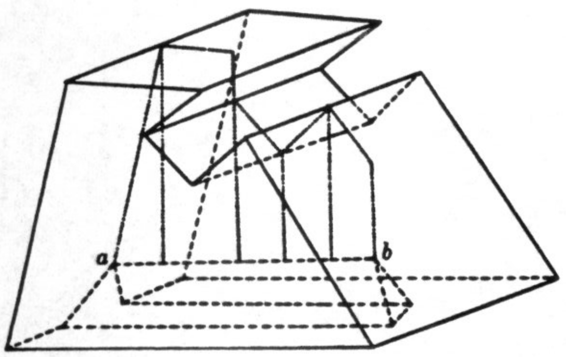

Figure 8: The fence of the edge ab

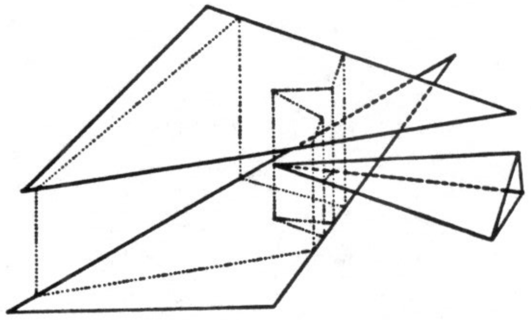
Figure 9: A nonconvex decomposition

partition P into cylindrical pieces, which are free of nonvertical reflex edges. Then, by triangulating the base polygons of each cylindrical piece, we can refine the partition into one consisting of cylindrical pieces whose base polygons are triangles. A triangulation of P follows trivially.

After all fences have been in, the partition of P involves a total of $O(n^2)$ vertices, edges, and facets. Triangulating the base polygons and finishing off the triangulation of P adds a constant multiplicative factor to the size of the decomposition. Therefore, the description size of the final partition is $O(n^2)$, and consequently $O(n^2)$ tetrahedra are produced. The execution time for the entire fence-off phase is $O(n^2 \log n)$.

B. The pop-out phase. This is a form of preprocessing aimed at bringing down the number of vertices of a given polytope P to the same order as the number of its reflex edges. Our attempt will be to remove the cups of a number of pointed vertices whose domes are not hindered. The boundary of the convex hull of v's crown, for any pointed vertex v of P, consists of two polyhedral patches separated by the crown itself, one of which isolates the other from v. Let K be the polytope bounded by the patch in question and the cone of v. The cup of v is precisely K if and only if every reflex vertex and edge of P lies either outside K or on the boundary of the cone. The direct implementation of the above test for hindered domes, however, involves checking each such K against each of the reflex vertices, and thus amounts to an undesirable $O(nr)$ time complexity for this phase. To improve on this, we first compute a superset of the pointed vertices that have hindered domes, which we keep appropriately updated during the removal of the cups in accordance with the local changes in the polytope. The following lemma provides us with the idea to carry out this computation.

Lemma 3.1. *Let v be a pointed vertex, and p any point of $dome(v) \setminus crown(v)$. Then, for every line passing through p, at least one of the two rays from p must intersect a facet of the triangulated boundary of the polytope incident upon v.*

Proof: By definition, the dome of v is a concave polyhedral patch with respect to its cup. Therefore, either at least one of the two rays from p intersects the interior of the cup, or the line lies on the dome, and thus intersects the crown (since in that case the dome consists of coplanar facets). In either case, the line intersects one of the facets of the cup incident upon v. For a given triangulation τ of ∂P, let $\pi(z, \tau)$ denote the connected polyhedral patch consisting of the facets of P incident upon z. It is easy to see that the facets of the cup of v that are incident upon the apex constitute the set intersection of all the patches $\pi(v, \tau_i)$ over all possible boundary triangulations τ_i. Thus, we conclude that the line intersects a facet of the polytope incident upon v. ∎

Note that any internal vertex of the dome of v, as well as any point of an internal edge outside its endpoints fits the above description of p. In particular, an internal vertex of a dome is provided by a reflex vertex p of the polytope P such that there exists a small open halfball centered at p that lies entirely in the interior of P. None of the other reflex vertices can hinder any dome. Additionally, a dome may be hindered by a reflex edge e of P that is internal to the dome and has both its endpoints on the crown. This case can be reduced to the previous one, provided that we introduce a dummy reflex vertex at the midpoint of e. Consequently, any hindered dome contains an internal reflex vertex, dummy or not, and therefore these vertices alone can determine the pointed vertices whose domes are hindered. At the outset of the pop-out phase, we assume that every reflex vertex as well as the midpoint of every reflex edge is potentially hindering a dome. Local tests on the incidence structures of the vertices might allow us to weed out many candidates. Although this might be a practical step to take, it is not necessary strictly speaking. The reflex vertices and the midpoints of the reflex edges are collectively called ***puncturing*** vertices, and their number is no more than $3r$.

As a result of the previous lemma, we can find a superset of the pointed vertices that have hindered domes, by considering vertical lines passing through all the puncturing vertices, and compiling a list of the vertices of the facets of P that these vertical lines intersect. For convenience, we assume that no vertex of the polytope lies vertically above or below any of the puncturing vertices. Note that this assumption can be checked in $O(n \log r)$ time and can be relaxed with little extra effort. So, our task is to determine which (triangular) facet of P is traversed by a vertical line passing through a puncturing vertex. To speed up this operation, we set up a complete binary tree whose leaves are in one-to-one correspondence, from left to right, with the puncturing vertices of P, ordered by nondecreasing x-coordinates. With each internal node v of the tree, we associate the canonical strip $\{(x,y) \,|\, a \leq x \leq b\}$, where a and b are respectively the smallest and largest x-coordinates of the vertices associated with the leaves descending from v.

The questions we have to answer now are of the form: Which projections of puncturing vertices (on the xy-plane) lie in a given triangle? We dualize the problem by mapping a point (u,v) to the line $ux + vy + 1 = 0$. In this way, to each node of the tree corresponds a certain arrangement of lines (the duals of the projections of the vertices stored at the leaves below the node in question). We do not store these arrangements explicitly, except the one at the root of the tree. This takes $O(r^2)$ time and space, using (Chazelle et al. [9], Edelsbrunner et al. [14]). We further process the root arrangement to support $O(\log r)$-time point location (Kirkpatrick [17], Edelsbrunner et al. [13]), which requires a linear amount of work through the arrangement. As it turns out, we also need point location structures for all the other arrangements in the tree. We can use an economical strategy, however, based on the fact that the arrangement of a node is a portion of the arrangement of its father. More specifically, each region of a father's arrangement lies entirely within one region of either child's. Thus, we provide each region of the root arrangement with two pointers, one directed toward the enclosing region for each child. The same pointer scheme can be applied throughout the tree, although the only arrangement, and hence geometric information, stored in the tree lies at the root and at the leaves. Setting up all these pointers can be done in $O(r^2)$ time by ensuring that the work at a given node is at most proportional to the square of the number of its descending leaves. In the general step, we have at our disposal the full arrangement at a given node, and we color the lines green or red, depending on which of the left or right children inherits them. To compute, say, the green arrangement with the pointers directed to it, we begin by merging collinear green edges into edges of the green arrangement. By traversing the full arrangement, we can now collect each region lying within a given region of the green arrangement. The same process applied to the red edges completes our work.

Let f be a triangular facet of P and let abc be its projection on the xy-plane, with $a \leq b \leq c$ in x-order. Our tree structure allows us to home in very quickly on the puncturing vertices whose projections lie within abc. First, we use the x-order of the leaves to partition the x-extents of ab and ac into $O(\log r)$ intervals, following standard segment tree partition. This, in turn, induces a partition of abc into a logarithmic number of trapezoids (or triangles), every one of which is associated with a distinct node of the tree. For each such node v, we must determine which points among those stored in v's descending leaves lie inside the corresponding trapezoid. One nice feature in this set-up is that although we have many different trapezoids, we can replace each of them in the computation by one of two fixed double wedges. In dual space, this problem is the same as computing which lines of the arrangement at v intersect one of two line segments. This is easily done using our data structure. First we locate the endpoints of the segment in the root arrangement, and then using region-to-region pointers, we go down the tree and deduce the location of the endpoints in the arrangements at the selected vertices. The entire operation takes only $O(\log r)$ time. Now we must solve the following problem: Given two points a and b located in the arrangement at node v, find all the intersections between ab and the arrangement. Here is a simple method: If a and b lie in the same region, no further work need be done. Otherwise, we pursue our search in the two children of v. This process takes us to a certain subset of the leaves, where the desired intersections can be found directly. Since a vertical line can cut only $O(r)$ facets, the total amount of time spent computing intersections between lines through puncturing vertices and facets of P is no more than $O\big(n \log r + r^2 \log r\big)$. Note that this is a one-shot cost, and the above process is executed only once during the pop-out phase.

Of all the intersections between a vertical line through a given puncturing vertex and the facets of P, only those closest to the puncturing vertex need to be retained for further consideration in light of Lemma 3.1. The corresponding facets are called *witnesses*. At this point of the pop-out phase, we have established the following invariant, which we will maintain throughout the rest of the phase: the witness facets constitute a collection of $O(r)$ facets of P whose vertices are a superset of the pointed vertices with hindered domes; the record of each such facet contains a list of the puncturing vertices whose vertical lines intersect the facet.

The main body of the pop-out phase is an iterative process. In each pass, we go through all the pointed vertices of the polytope, looking for those of degree below some appropriate constant d, and we insert them in a queue of *favorable* vertices. Once all vertices have been processed, we repeat the following steps until the queue is empty.

1. Let v be the favorable vertex referenced by the top of the queue. Compute the convex hull H of v's crown, retriangulating the boundary of P at the same time, so that the resulting polyhedral patch formed by the facets of P incident upon v is precisely $\big(\partial cup(v)\big) \setminus dome(v)$. Any information associated with facets of P that are affected is updated to reflect any changes.

2. If v is incident upon any witness facet f, check whether the puncturing vertices associated with f lie outside H, thus determining whether the dome of v is hindered or not. Since the cup of v is of constant size, this can be done in time proportional to the number of puncturing vertices probed. If the dome of v is found hindered, remove v from the queue, and proceed with step 1. Otherwise, the new witness facets that result upon f's removal need be determined. These are the facets of the dome of v that lie vertically above or below f, and they can be found by

scanning H. Subsequently, lists of puncturing vertices associated with them are appropriately set up.

3. Triangulate v's cup by connecting it to each nonincident edge of its triangulated boundary, and remove from P each tetrahedron obtained in this way, renormalizing the resulting polytope.
4. Remove v from the queue, and mark its adjacent vertices, so that they are skipped if met in the queue later.

Processing all favorable vertices along these four steps takes a total of $O(dn_i)$ time, where n_i is the size of the polytope at the current pass. From Lemma 2.7 and the fact that at most $d+1$ favorable vertices are marked or removed from the queue each time step 4 is executed, we derive that this process removes at least

$$\frac{1}{d+1}\,\frac{(1-c)(d-5)}{(d-2)}n_i \geq \alpha n_i$$

vertices from the polytope, where α is a fixed positive constant less than 1. Thus we are left with a polytope with $n_{i+1} \leq (1-\alpha)n_i$ vertices and at most r reflex edges. Repeating this pruning pass until $n_i = O(r)$ takes time proportional to $\sum n_i = O(n)$. Note that as vertices are being popped out, some reflex vertices may become pointed. Therefore, the set of puncturing vertices, and consequently the set of witness facets, may need updating. We choose, however, to make no updates at all. Although some unnecessary work may be done, the stated time complexity is not affected.

Summarizing, the pop-out phase accounts for $O(n\log r + r^2\log r)$ in the running time of the algorithm. The total space needed amounts to $O(n + r^2)$. Finally, since each popped-out vertex produces at most $d-2$ new tetrahedra, by the end of this phase, P is decomposed into a collection of $O(n)$ tetrahedra and a polytope of $O(r)$ vertices.

C. Putting the pieces together. Given a simple polytope with n vertices and r reflex edges, we start the partitioning by (i) removing all flat vertices, and doing the obvious clean-up, (ii) triangulating the boundary, and (iii) applying the pop-out phase in case n greatly exceeds r. We finish the decomposition by going through the fence-off phase. The running time of the algorithm is $O(n\log r+r^2\log r)$. In practice, it will be important to have a robust representation of cell complexes in 3-space in order to carry out the computation successfully and efficiently. A representation of three-dimensional polyhedral subdivisions, along with the set of navigational primitives needed to carry out the required cutting operations, can be found in (Dobkin and Laszlo [12]). We summarize our results below.

Theorem 3.1. *In $O((n+r^2)\log r)$ time it is possible to partition a simple polytope with n vertices and r reflex edges into $O(n+r^2)$ tetrahedra. The time bound includes the cost of producing a full-fledged triangulation with an explicit description of its facial structure. Up to within a constant factor, the number of tetrahedra produced by the algorithm is optimal in the worst case.*

4. Closing Remarks

Of course, not every n-vertex polytope with r reflex edges necessitates $\Omega(n+r^2)$ tetrahedra to form a triangulation. Are there simple heuristics one could use to guarantee that the triangulation size does not exceed the minimum by more than a fixed constant in *all* cases? Is there a polynomial-time algorithm for such an approximation scheme? Also, it is often desirable to avoid long, skinny tetrahedra in mesh generation. See (Baker et al. [4]) for similar concerns in two dimensions. One approach is to retriangulate the undesirable tetrahedra produced by our triangulation. Again, are there preferred heuristics to keep the number of Steiner points as low as possible?

REFERENCES

1. Aronov, B., Sharir, M. *Triangles in space, or building and analyzing castles in the air*, Proc. 4th Ann. ACM Sympos. Comput. Geom. (1988), 381-391.
2. Bajaj, C.L., Dey, T.K. *Robust decompositions of polyhedra*, Dept. Comp. Science, Purdue Univ. 1989.
3. Baker, T.J. *Three-dimensional mesh generation by triangulation of arbitrary point sets*, Dept. Mechanical Engineering, Princeton Univ. 1986.
4. Baker, B.S., Grosse, E., Rafferty, C.S. *Non-obtuse triangulation of polygons*, Discrete and Computat. Geom. 3 (1988), 147–168.
5. Baumgart, B.G. *A polyhedron representation for computer vision*, 1975 National Computer Conference, AFIPS Conf. Proceedings, 44, AFIPS Press (1975), 589–596.
6. Canny, J.F. *A new algebraic method for motion planning and real geometry*, Proc. 28th Annu. IEEE Symp. on Foundat. of Computer Science (1987), 39–48.
7. Chazelle, B. *Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm*, SIAM J. Comput. 13 (1984), 488–507.
8. Chazelle, B. *Approximation and decomposition of shapes*, Advances in Robotics, Vol.1: Algorithmic and Geometric Aspects of Robotics, (J.T. Schwartz and C.K. Yap, ed.), Lawrence Erlbaum Associates (1987), 145–185.
9. Chazelle, B., Guibas, L.J., Lee, D.T. *The power of geometric duality*, BIT 25 (1985), 76–90.
10. Chazelle, B., Edelsbrunner, H., Guibas, L.J., Sharir, M. *A singly-exponential stratification scheme for real semi-algebraic varieties and its applications*, Proc. 16th ICALP, Springer-Verlag, LNCS 372 (1989), 179–193.
11. Collins, G.E. *Quantifier elimination for real closed fields by cylindric algebraic decomposition*, Proc. 2nd GI Conf. on Automata Theory and Formal Languages, Springer-Verlag, LNCS 33 (1975), 134–183.
12. Dobkin, D.P., Laszlo, M.J. *Primitives for the manipulation of three-dimensional subdivisions*, Proc. 3rd Ann. ACM Sympos. Comput. Geom. (1987), 86–99.
13. Edelsbrunner, H., Guibas, L.J., Stolfi, J. *Optimal point location in a monotone subdivision*, SIAM J. Comput. 15 (1986), 317–340.
14. Edelsbrunner, H., O'Rourke, J., Seidel, R. *Constructing arrangements of lines and hyperplanes with applications*, SIAM J. Comput. 15 (1986), 341–363.
15. Feng, H., Pavlidis, T. *Decomposition of polygons into simpler components: Feature generation for syntactic pattern recognition*, IEEE Trans. Comp., C–24 (1975), 636-650.
16. Guibas, L.J., Stolfi, J. *Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams*, ACM Trans. Graphics 4 (1985), 75–123.
17. Kirkpatrick, D.G. *Optimal search in planar subdivisions* SIAM J. Comput. 12 (1983), 28–35.
18. Lingas, A. *The power of non-rectilinear holes*, Proc. 9th Colloquium on Automata, Languages and Programming, Aarhus, LNCS Springer-Verlag (1982), 369–383.

19. Mehlhorn, K. *Data Structures and Algorithms 3: Multidimensional Searching and Computational Geometry*, Springer-Verlag, Heidelberg, Germany, 1984.
20. Muller D.E., Preparata, F.P. *Finding the intersection of two convex polyhedra*, Theoret. Comput. Sci. 7 (1978), 217–236.
21. O'Rourke, J. *Art gallery theorems and algorithms*, Oxford Univ. Press, 1987.
22. Prill, D. *On approximations and incidence in cylindrical algebraic decompositions*, SIAM J. Comput. 15 (1986), 972–993.
23. Ruppert, J., Seidel, R. *On the difficulty of tetrahedralizing 3-dimensional non-convex polyhedra*, Proc. 5th Ann. ACM Sympos. Comput. Geom. (1989), 380-392.
24. Schachter, B. *Decomposition of polygons into convex sets*, IEEE Trans. Comp., C–27 (1978), 1078–1082.
25. Schwartz, J.T., Sharir, M. *On the "piano movers" problem. II: General techniques for computing topological properties of real algebraic manifolds*, Adv. in Appl. Math. 4 (1983), 298–351.
26. Smith, W. *Studies in computational geometry motivated by mesh generation*, PhD Thesis, Princeton Univ. 1988.
27. Whitney, H. *Elementary structure of real algebraic varieties*, Annals of Math. 66 (1957).