REPORT R-682 APRIL, 1975 UILU-ENG 75-2217

COORDINATED SCIENCE LABORATORY

CONVEX HULLS OF FINITE PLANAR AND SPATIAL SETS OF POINTS

F.P. PREPARATA
S.J. HONG

UNIVERSITY OF ILLINOIS – URBANA, ILLINOIS

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) CONVEX HULLS OF FINITE PLANAR AND SPATIAL SETS OF POINTS	5. TYPE OF REPORT & PERIOD COVERED Technical Report
	6. PERFORMING ORG. REPORT NUMBER R-682; UILU-ENG 75-2217
7. AUTHOR(s) F. P. Preparata and S. J. Hong	8. CONTRACT OR GRANT NUMBER(s) DAAB-07-72-C-0259
9. PERFORMING ORGANIZATION NAME AND ADDRESS Coordinated Science Laboratory University of Illinois at Urbana-Champaign Urbana, Illinois 61801	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Joint Services Electronics Program Fort Monmouth, New Jersey	12. REPORT DATE April, 1975
	13. NUMBER OF PAGES 21
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Convex hull
Computational complexity
Planar set of points
Spatial set of points
Recurrence

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The convex hulls of planar and spatial sets of n points can be determined with 0(n lg n) operations. The presented algorithms use the "divide and conquer" technique and recursively apply a merge procedure for two non-intersecting convex hulls. It is also shown that any convex hull algorithm requires at least 0(n lg n) operations, so that the time complexity of the proposed algorithms is optimal within a multiplicative constant.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

UILU-ENG 75-2217

CONVEX HULLS OF FINITE PLANAR
AND SPATIAL SETS OF POINTS

by

F. P. Preparata and S. J. Hong

This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy and U.S. Air Force) under Contract DAAB-07-72-C-0259.

CONVEX HULLS OF FINITE PLANAR AND SPATIAL SETS OF POINTS[†]

F. P. Preparata and S. J. Hong*
Coordinated Science Laboratory and Department of Electrical Engineering
University of Illinois at Urbana-Champaign

April 30, 1975

Abstract

The convex hulls of planar and spatial sets of n points can be determined with O(n lg n) operations. The presented algorithms use the "divide and conquer" technique and recursively apply a merge procedure for two nonintersecting convex hulls. It is also shown that any convex hull algorithm requires at least O(n lg n) operations, so that the time complexity of the proposed algorithms is optimal within a multiplicative constant.

---

*On leave from IBM, Systems Product Division, Poughkeepsie, N. Y.

[†]This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy, and U.S. Air Force) under Contract DAAB-07-72-C-0259.

1. Introduction

The determination of the convex hull of a finite set of points is relevant to several problems in computer graphics, design atuomation, pattern recognition and operation research: references [1][2][3]--just to cite a few--discuss some interesting applications in these areas which require convex hull determination.

Two relatively recent papers [4] [5] have considered the problem of determining the convex hull of a finite set of n points in the plane. R. L. Graham [4] described an algorithm based on representing the points in polar coordinates and sorting them according to their azimuth; the corresponding number of operations was shown to be at most $n \lg n + Cn$(1), for some constant C determined by the cartesian-to-polar coordinate conversion. Subsequently R. A. Jarvis [5] presented an alternative algorithm, which avoids coordinate conversions but has a running time $O(nm)$, where m is the number of points in the convex hull, claiming the superiority of his algorithm for small m.

In this paper we show that the convex hull of a planar set of points can be determined with at most $O(n \lg n)$ operations without resorting to coordinate conversions. We shall also show that the technique is generalizable to spatial sets of points, still maintaining the same order of complexity. Since the methods are based on the fact that the number of edges of the convex hull of n points is at most linear in n, its generalization does not seem possible beyond three dimensions. In fact when the number of dimensions

---

(1) "lg" denotes $\log_2$.

is no smaller than 4, it is known that there exist convex polyhedra with n vertices whose numbers of edges are $O(n^2)$ (see [6], p. 193).

Our algorithms are based on the well-known technique called "divide and conquer". Specifically let V be a d-dimensional Euclidean space (here, d = 2, 3) and let the set $S = \{a_1, \ldots, a_n | a_j \in V\}$ be given. By $x_i(a)$ we denote the i-th coordinate of $a \in V$, for $i = 1, \ldots, d$. Here and hereafter we assume that for any two points u and v in V we have $x_i(u) \neq x_i(v)$, for $i = 1, \ldots, d$. This simplification helps bring out the basic ideas of the algorithms to be described, while the modifications required for the unrestricted case are straightforward.

As a preliminary step we sort the elements of S according to the coordinate x_1, and relabel them if necessary so that we may assume $x_1(a_i) < x_1(a_j) \Leftrightarrow i < j$. We can now give the following algorithm:

<u>Algorithm CH</u>

<u>Input</u>: A set $S = \{a_1, \ldots, a_n\}$, where $a_j \in V$ and $x_1(a_i) < x_1(a_j)$ $\Leftrightarrow i < j$ for $i, j = 1, \ldots, n$.

<u>Output</u>: The convex hull CH(S) of S.

<u>CH1</u>. Subdivide S into $S_1 = \{a_1, \ldots, a_{\lfloor n/2 \rfloor}\}$ and $S_2 = \{a_{\lfloor n/2 \rfloor + 1}, \ldots, a_n\}$.
<u>CH2</u>. Apply recursively Algorithm CH to S_1 and S_2 to obtain $CH(S_1)$ and $CH(S_2)$.
<u>CH3</u>. Apply a <u>merge</u> algorithm to $CH(S_1)$ and $CH(S_2)$ to obtain CH(S) and halt.

The initial sorting of the elements of S requires $O(n \lg n)$ operations. Notice that, because of this sorting, the sets $CH(S_1)$ and $CH(S_2)$ will define two nonintersecting convex domains. Now, if the merging of two convex hulls with at most n d-dimensional extreme points in total requires at most

$P_d(n)$ operations, an upper-bound to the number $C_d(n)$ of operations required by Algorithm CH is given by the equation

$$C_d(n) = 2C_d(n/2) + P_d(n).$$

(Notice that we have assumed that n be even for simplicity, but practically without loss of generality). Thus, if we can show that $P_d(n)$ is $O(n)$, we shall obtain that $C_d(n)$ is $O(n \lg n)$, and, taking into account the initial sorting pass, an overall complexity $O(n \lg n)$ results for the convex hull determination.

In Sections 3 and 4 we shall show that merging algorithms with number of operations $O(n)$ can be designed for $d = 2, 3$. In the next section we shall establish a lower-bound to the number of operations performed by any algorithm for finding the convex hull of a set of n points. Since this computational work is a least of the same order as that of an algorithm for sorting n numbers, i.e., it is $O(n \lg n)$, we reach the interesting conclusion that the proposed convex hull algorithms for planar and spatial sets are optimal on their order of complexity, within a multiplicative constant.

2. A lower-bound.

The arguments presented in this section are similar to those developed in connection with finding the maxima of a set of vectors (see [7], Section 2), which is a problem only superficially related to the one being presently investigated. We begin with the following simple observation.

Lemma 1. $C_{d-1}(n) \leq C_d(n)$ for $d \geq 3$.

Proof: Let A_{d-1} be a set of n (d - 1)-dimensional points for $d \geq 3$, and let A_d be the set of d-dimensional points obtained by extending each point $v \in A_{d-1}$ with the same component v_d. Let $CH(A_d)$ and $CH(A_{d-1})$ be the convex hulls of A_d and A_{d-1}. Clearly the projection of $CH(A_d)$ on the coordinates $x_1, \ldots, x_{d-1}$ is $CH(A_{d-1})$. Thus to find $CH(A_{d-1})$, it suffices to find $CH(A_d)$, whence $C_{d-1}(n) \leq C_d(n)$.

Lemma 2. $C_2(n) \geq O(n \lg n)$ for $n \geq 3$.

Proof: Let $A = \{a_1, \ldots, a_n\}$ be a planar set of points, and assume that $CH(A) = A$: this means that the points $a_1, \ldots, a_n$ are the vertices of a convex polygon and may be thought of as forming a circuit. There are four points in A, a_{j_0}, a_{j_1}, a_{j_2}, and a_{j_3} such that $x_2(a_{j_0}) = \max_i x_2(a_i)$,

$$x_1(a_{j_1}) = \max_i x_1(a_i),\ x_2(a_{j_2}) = \min_i x_2(a_i) \text{ and } x_1(a_{j_3}) = \min_i x_1(a_i)$$

(see figure 1). Considering these four points as a cyclic sequence, there

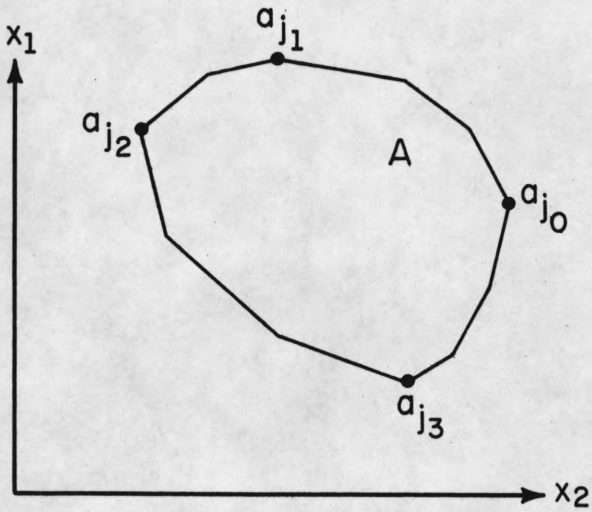


Figure 1. Illustration for the proof of Lemma 2.

is one pair of consecutive elements in this sequence which comprises at least ($\lceil n/4 \rceil$ - 2) points of A. Without loss of generality, let this pair be (a_{j_0}, a_{j_i}), and let $a_{j_0} = a_1', a_2', \ldots, a_s' = a_{j_1}$ be the sequence of vertices comprised between a_{j_0} and a_{j_1}, with $s \geq \lceil n/4 \rceil$. For any three points, a, b, and c in the plane we define $M_3(a,b,c)$, $M_2(a,b,c)$, and $M_1(a,b,c)$ as follows:

$$x_2(M_3(a,b,c)) = \text{maximum } \{x_2(a),\ x_2(b),\ x_2(c)\}$$
$$x_2(M_2(a,b,c)) = \text{median } \{x_2(a),\ x_2(b),\ x_2(c)\}$$
$$x_2(M_1(a,b,c)) = \text{minimum } \{x_2(a),\ x_2(b),\ x_2(c)\}.$$

Consider now any algorithm for finding the convex hull of a planar set. For any triplet (a_i', a_j', a_k'), with i, j, and k in the range [1,s], the algorithm must be able to decide whether or not the following convexity condition holds (where $M_\ell = M_\ell(a_i', a_j', a_k')$, for ℓ = 1, 2, 3)

$$x_1(M_2) < [(x_2(M_2) - x_2(M_3)) \cdot x_1(M_1) + (x_2(M_1) - x_2(M_2)) \cdot x_1(M_3)] / (x_2(M_1) - x_2(M_3)).$$

This implies that the algorithm must also be able to decide, for any three points a_i', a_j', and a_k', the relative ordering of their x_2 coordinates, which is equivalent to saying that the ordering of the coordinates $x_2(a_1')$, $x_2(a_2')$, ..., $x_2(a_s')$ must be known. As is well-known, this requires a number of operations equivalent to at least $\lceil \lg s! \rceil$ comparisons. Recalling that $s \geq n/4$, we have $\lceil \lg s! \rceil \geq \lceil \lg(n/4)! \rceil \approx \frac{n}{4} \lg \frac{n}{4} - O(n)$, whence $C_2(n) \geq O(n \lg n)$.

We obtain the following conclusion:

<u>Theorem</u>. $C_d(n) \geq C_{d-1}(n) \geq \ldots \geq C_2(n) \geq O(n \lg n)$.

3. A merge algorithm for planar sets.

Let $A = (a_1, \ldots, a_p)$ and $B = (b_1, \ldots, b_q)$ be two planar convex polygons, where $(a_1, \ldots, a_p)$ is the (clockwise) sequence of the vertices in the perimeter of A, and similarly is $(b_1, \ldots, b_q)$ for B. We assume that $x_1(a_i) < x_1(b_j)$ for $i = 1, \ldots, p$ and $j = 1, \ldots, q$, so that A and B are nonintersecting.

By merging A and B we mean the determination of the convex hull CH(A,B) of A and B. The convex polygon CH(A,B) is obtained by tracing the two tangents common to A and B and by eliminating the points of A and B which become internal to the resulting polygon (see figure 2).

We let ℓ_A and r_A be two points of A such that $x_2(\ell_A) = \min_i x_2(a_i)$ and $x_2(r_A) = \max_i x_2(a_i)$; similarly ℓ_B and r_B are defined in B. For easy reference, we shall call the two tangents to A and B as left and right tangent. It is easily realized that the determination of, say, the right tangent depends upon the relative ordering of $x_2(r_A)$ and $x_2(r_B)$; the same can be said for the left tangent in relation to $x_2(\ell_A)$ and $x_2(\ell_B)$. Therefore in the sequel we shall consider only one case, specifically the determination of the right tangent under the hypothesis

$$x_1(r_A) < x_1(r_B) \quad \text{and} \quad x_2(r_A) < x_2(r_B);$$

the other case, as well as the determination of the left tangent, are treated in an analogous manner. Without loss of generality, we shall also assume that $r_A = a_1$ and $r_B = b_1$. Indices of vertices of A and B are assumed to be taken mod p and mod q, respectively.

Given two points u and v in the plane, (u,v) and $\overline{(u,v)}$ denote respectively the line containing u and v and the segment delimited by u and v. The slope $sl(u,v)$ is given by $sl(u,v) = (x_1(u) - x_1(v))/(x_2(u) - x_2(v))$.

We must now determine the two vertices a_{i*} of A and b_{j*} of B which are the extremes of the right tangent, where $1 \leq i* \leq index[\ell_A]$ and $1 \leq j* \leq index[\ell_B]$. We begin by defining the slopes:

$$\alpha_{i,i+1} = sl(a_i, a_{i+1}),\quad \beta_{j,j+1} = sl(b_{j,j+1}),\quad \gamma_{ij} = sl(a_i, b_j).$$

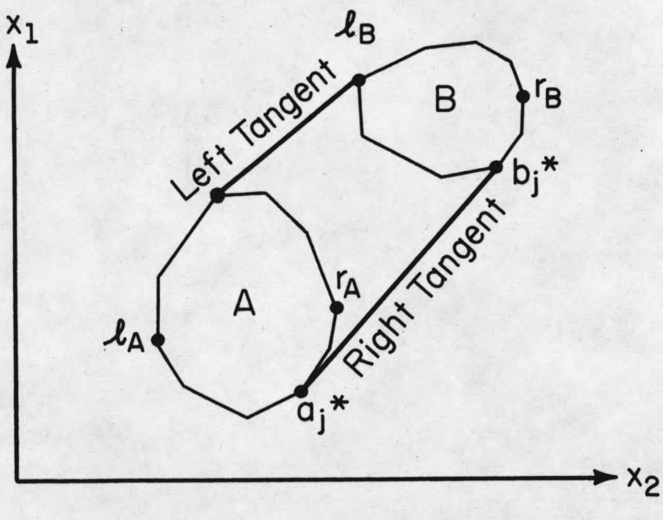


Figure 2. Illustration of the planar merge procedure.

Notice that in the ranges $1 \leq i < index[\ell_A]$ and $1 \leq j < index[\ell_B]$, due to convexity, the sequences $(\alpha_{12}, \alpha_{23}, \ldots\ )$ and $(\beta_{12}, \beta_{23}, \ldots\ )$ are strictly monotone decreasing. Thus, the extremes a_{j*} and b_{j*} of the right tangent are characterized by the following properties:

$$(1)\begin{cases} i^* > 1 \Rightarrow \alpha_{i^*,i^*+1} < \gamma_{i^*j^*} \le \alpha_{i^*-1,i^*};\ j^* \ge 1 \Rightarrow \beta_{j^*,j^*+1} \le \gamma_{i^*j^*} < \beta_{j^*-1,j^*} \\ i^* = 1 \Rightarrow \qquad \alpha_{12} < \gamma_{1j^*} \qquad\qquad ;\ j^* = 1 \Rightarrow \qquad \beta_{12} \le \gamma_{i^*1}. \end{cases}$$

We claim that the following algorithm uniquely determines a_{i^*} and b_{j^*}.

<u>Algorithm RT (right tangent)</u>

<u>Input</u>: Coordinates of $(a_1, a_2, \ldots, \ell_A)$ and $(b_1, b_2, \ldots, \ell_B)$, and slopes $(\alpha_{12}, \alpha_{23}, \ldots)$ and $(\beta_{12}, \beta_{23}, \ldots)$.

<u>Output</u>: i^*, j^*, the indices of the extremes of the right tangent segment.

<u>RT1</u>. Set $i \leftarrow 1$, $j \leftarrow 1$.

<u>RT2</u>. Compute $\gamma_{ij} \leftarrow (x_1(a_i) - x_1(b_j))/(x_2(a_i) - x_2(b_j))$.

<u>RT3</u>. If $\alpha_{i,i+1} \ge \gamma_{ij}$, set $i \leftarrow i + 1$ and go to RT2.

<u>RT4</u>. If $\beta_{j,j+1} > \gamma_{ij}$, set $j \leftarrow j + 1$ and go to RT2.

<u>RT5</u>. Set $i^* \leftarrow i$, $j^* \leftarrow j$, and halt.

We now prove the validity of Algorithm RT. The algorithm halts when the conditions $\alpha_{i,i+1} < \gamma_{j,j+1}$ and $\beta_{j,j+1} \le \gamma_{ij}$ occur for the first time. Thus all we have to show is that before executing step RT3 we always have $\gamma_{ij} \le \alpha_{i-1,i}$ and $\gamma_{ij} < \beta_{j-1,j}$. We distinguish two cases: (1) j is incremented or (2) i is incremented.

(1) The index j is incremented when the condition $\alpha_{i,i+1} < \gamma_{ij} < \beta_{j,j+1}$ occurs. Assuming inductively that $\gamma_{ij} \le \alpha_{i-1,i}$ we have (see figure 3a)

$\gamma_{i,j+1} \leq \gamma_{ij} \leq \alpha_{i-1,i}$ and $\gamma_{i,j+1} < \beta_{j,j+1}$: after incrementing j, these conditions become $\gamma_{ij} \leq \alpha_{i-1,i}$ and $\gamma_{ij} < \beta_{j-1,j}$, as desired.

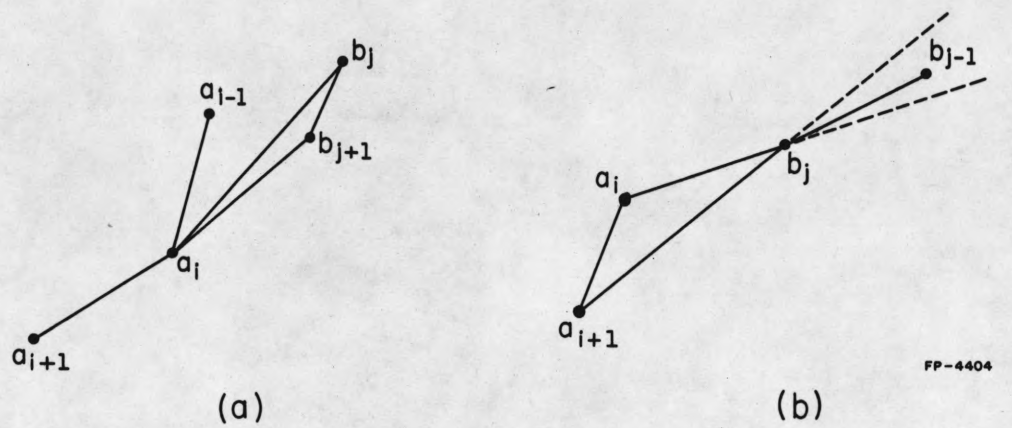


Figure 3. Illustrations for the validity of Algorithm RT.

(2) The index i is incremented when $\gamma_{ij} \leq \alpha_{i,i+1}$. Notice at first that we cannot have $\beta_{j-1,j} \leq \gamma_{i+1,j}$: indeed $\beta_{j-1,j} \leq \gamma_{i+1,j}$ implies $\alpha_{i,i+1} > \gamma_{i,j-1}$, whence, by the formulation of step RT3, the vertex b_j cannot have been reached (see figure 3b) yet by the algorithm; thus we have $\gamma_{i+1,j} < \beta_{j-1,j}$. Next, we notice that when $\gamma_{ij} \leq \alpha_{i,i+1}$, we also have $\gamma_{i+1,j} \leq \alpha_{i,i+1}$. The two conditions $\gamma_{i+1,j} < \beta_{j-1,j}$ and $\gamma_{i+1,j} \leq \alpha_{i,i+1}$ become $\gamma_{ij} < \beta_{j-1,j}$ and $\gamma_{ij} \leq \alpha_{i-1,i}$ after incrementing the index i, thus proving our original claim and the validity of the algorithm. It is clear that the number of operations performed by Algorithm RT is $O(i^* + j^*)$.

A procedure analogous to Algorithm RT is required for the determination of the other tangent to A and B (left tangent); clearly, the overall number of operations necessary for determining the two tangents is at most of order $(p + q)$. Finally, we recall that the data structure describing a convex polygon is simply a list giving the circular sequence of its vertices. Thus it is easily realized that the construction of the data structure describing CH(A,B) from the analogous data structures of A and B can be accomplished by modifying a fixed number (two) of pointers. Thus, the overall running time $P_2(n)$ of the merge algorithm of planar sets is at most linear in the total number n of vertices.

4. A merge algorithm for spatial sets.

The merge algorithm for planar sets described in the preceding section can be viewed as constructing a two-dimensional cylinder tangent to two given convex polygons. This idea is the basis for the three-dimensional procedure, which we shall now informally describe.

Let A and B be two convex polyhedra with p and q vertices, respectively. Again, we assume that for any points a_i of A and b_j of B we have $x_1(a_i) < x_1(b_j)$, so that A and B are nonintersecting.

It is a crucial observation that the sets of vertices and edges of either A or B form a planar graph: specifically, barring degeneracies, we may assume they form a triangulation. Thus we know that the numbers of edges of A and B are at most (3p - 6) and (3q - 6), respectively, by Euler's theorem (see, e.g., [6], p. 189).

The convex hull CH(A,B) of A and B may be obtained by the following operations (see figure 4 for an intuitive illustration):

1) Construction of a "cylindrical" triangulation $\mathcal{T}$, which is tangent to A and B along two circuits E_A and E_B, respectively.
2) Removal both from A and from B of the respective portions which have been "obscured" by $\mathcal{T}$.

Here, the terms "cylindrical" and "obscured" have not been formally defined; rather, they have been used in their intuitive connotations, as suggested by figure 4.

We begin by discussing the construction of the triangulation $\mathcal{T}$. The initial step is the determination of one edge of $\mathcal{T}$. This is easily done by projecting the polyhedra A and B on the plane $\langle x_1, x_2\rangle$ (see figure 4):

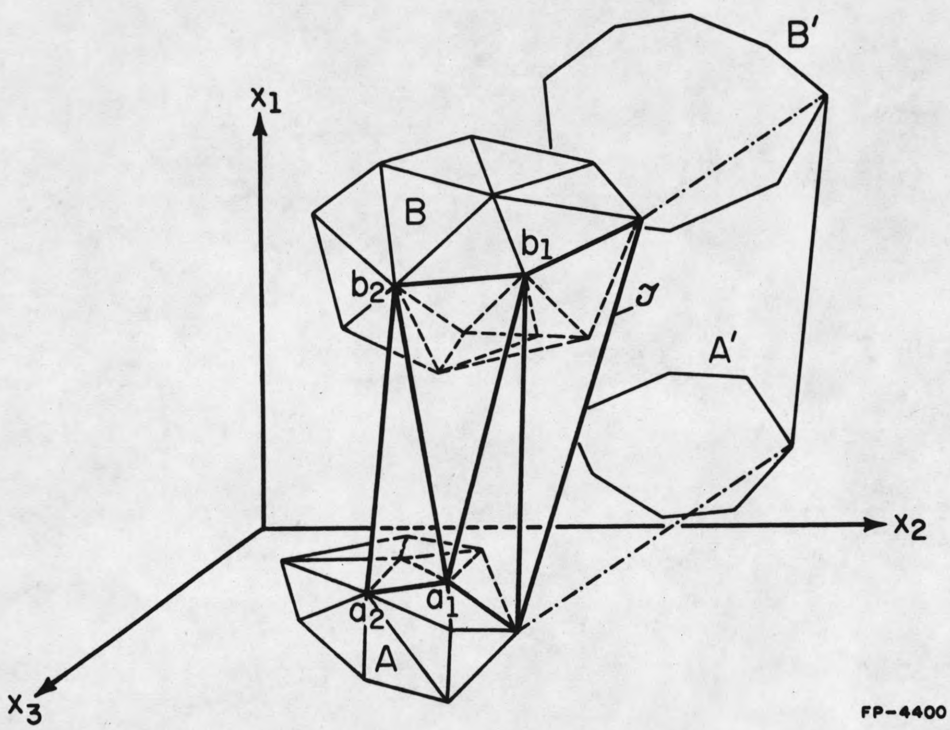


Figure 4. Merging two convex hulls. Construction of $\mathcal{T}$.

let A' and B' be the projections on $\langle x_1, x_2\rangle$ of A and B, respectively (obviously A' and B' are nonintersecting). We now apply the merge algorithm for planar sets, described in Section 2, to A' and B'. This operation yields one segment tangent to A' and B', whose extreme points are the projections of the extreme points of an edge of $\mathcal{T}$. Thus an edge of $\mathcal{T}$ has been determined and the construction can be started.

We shall now describe the advancing mechanism of the procedure, which determines at each step a new vertex of $\mathcal{T}$, thereby adding a new face to $\mathcal{T}$. In our illustration (figure 4), a_2 and (a_2, b_2, a_1) are, respectively, the vertex and the face of $\mathcal{T}$ constructured in the previous step. The advancing mechanism makes reference to the most recently constructed face of $\mathcal{T}$. To initialize the procedure, the reference face is chosen as one of the half planes parallel to the x_3 axis, containing the initially determined edge and delimited by it. Let (a_2, b_2, a_1) be the reference face for the current step. We must now select a vertex $\hat{a}$, connected to a_2, such that the face $(a_2, b_2, \hat{a})$ forms the largest convex angle with (a_2, b_2, a_1) among the faces (a_2, b_2, v), for all $v \neq a_1$ connected to a_2; similarly we select $\hat{b}$ among the vertices connected to b_2. For reasons to become apparent later, we call these comparisons of <u>type 1</u>.

Next, once the "winners" $(a_2, b_2, \hat{a})$ and $(a_2, b_2, \hat{b})$ have been selected, we have a run-off comparison, called of <u>type 2</u>. If $(a_2, b_2, \hat{a})$ forms with (a_2, b_2, a_1) a larger convex angle than $(a_2, b_2, \hat{b})$, then $\hat{a}$ is added to $\mathcal{T}$ ($\hat{b}$ is added in the opposite case) and the step is complete. Practically, the triangulation $\mathcal{T}$ is entirely specified by the circular sequence E_{AB} of the vertices which are successively acquired by the advancing mechanism just illustrated. In fact, this sequence E_{AB} is some interleaving of the two sequences of vertices of E_A and E_B; the interleaving exactly specifies the edges of $\mathcal{T}$ not belonging to E_A or E_B (see figure 5).

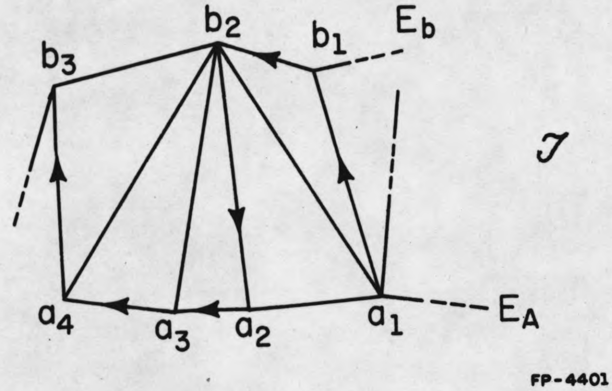


Figure 5. A fragment of $\mathcal{J}$ described by the string $a_1 b_1 b_2 a_2 a_3 a_4 b_3$.

To efficiently implement the outlined step, we make the following considerations. First we describe a criterion for uniquely ordering the edges incident on any vertex of A or B. For any a in A (b in B) the edges incident on a (on b) are numbered in ascending order so that they form a counterclockwise (clockwise in B) sequence for an external observer. For concreteness of illustration, suppose now that b and $\overline{(b, a)}$ are the most recently added vertex and edge of $\mathcal{J}$, respectively, and let $\overline{(b_1, b)}$ be the edge of E_B reaching b (see figure 5). Without loss of generality, we may assume that the numbering of the edges incident on b and of their terminals $b_1, b_2, \ldots, b_k$ be as shown in figure 6, where $k = 7$. Let (b_s, b, a) be the face which forms the smallest convex angle with (b_1, b, a) among the faces (b_i, b, a) for $i = 2, \ldots, k$ (in our case, $s = 4$). It is clear that any b_i for $1 < i < s$ is an internal point of the final hull CH(A,B) and need not be further considered.

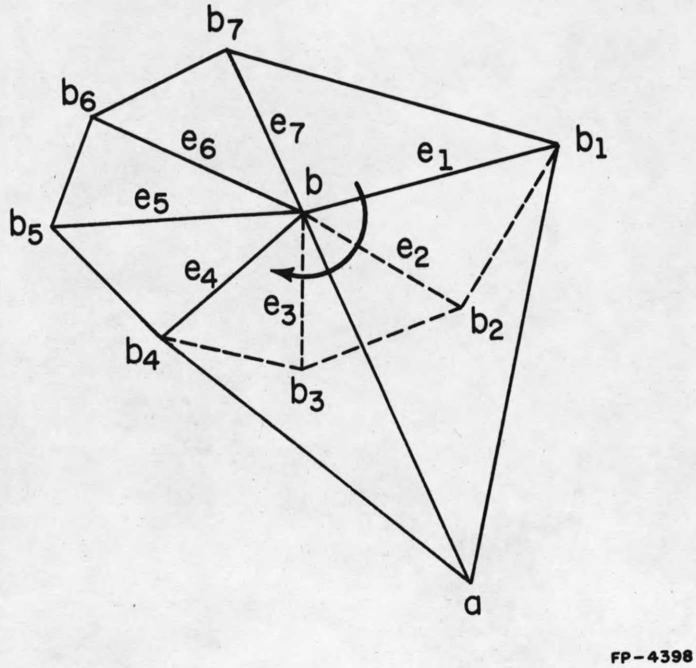
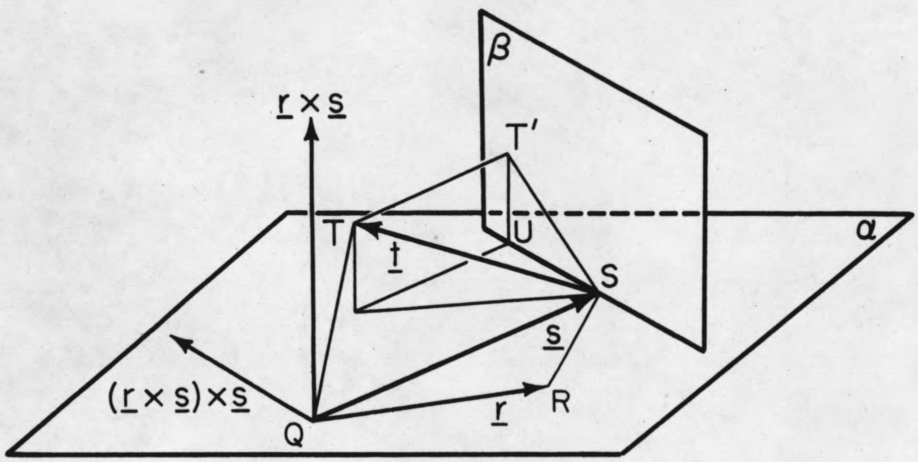


Figure 6.

Thus, we can easily upper-bound the number of comparisons of angles between pairs of planes required by the construction of $\mathcal{T}$. First of all, we notice that each type-1 comparison definitively eliminates one edge of either A or B from those considered by the procedure which constructs $\mathcal{T}$. Since the numbers of edges of A and B are at most (3p - 6) and (3q - 6), respectively, the number of type-1 comparisons is bounded by

$[(3p - 6) - 1 + (3q - 6) - 1] = 3(p + q) - 14$. Next, each type-2 comparison adds a new vertex to either E_A or E_B: since the numbers of vertices of E_A and E_B are at most p and q, respectively, the number of type-2 comparisons is bounded by $(p + q - 1)$. We conclude that the number of angle comparison grows no faster than linearly in the total number of vertices of A and B. Notice that this result rests crucially on the property that the numbers of edges of A and B are linear in their respective numbers of vertices.

It is now worth considering the implementation of the operation of comparing two angles, which is central to the outlined algorithm. We first notice that, due to convexity, all angles to be considered belong to the range $[0,\pi]$. Referring now to figure 7, consider the convex angle formed by the face QST with the face QRS, lying in plane α. Let β be a plane orthogonal to $\overline{RS}$ and T' be the projection of T on β: $(\pi - T'\hat{S}U)$ is the angle between QST and QRS. Since the function cotangent: $[0,\pi] \rightarrow [-\infty, +\infty]$ is an order-reversing mapping, we shall replace the comparison of two angles with the comparison of their cotangents, thereby avoiding costly computations of inverse trigonometric functions. Thus we must compute $\cot(\pi - T'\hat{S}U) = -\cot(T'\hat{S}U) = \overline{SU}/\overline{T'U}$. We shall use vector notation and let "$\times$" and "o" denote "outer" and "inner" products of 3-dimensional vectors, respectively; also, we let $\overline{QS} = \underline{s}$, and $\overline{SU} = \underline{t}$. Referring to figure 5, it is obvious that $\overline{SU} = K_1 \underline{t} \circ ((\underline{r} \times \underline{s}) \times \underline{s})$ and $\overline{T'U} = -K_2 \underline{t} \circ (\underline{r} \times \underline{s})$, where $K_1^{-1} = |\underline{r}| \cdot |\underline{s}|^2 \sin\theta$



FP-4399

Figure 7. Illustration of the cotangent calculation.

and $K_2^{-1} = |\underline{r}| \cdot |\underline{s}| \sin\theta$, θ being the angle between $\underline{r}$ and $\underline{s}$. It follows that $\overline{SU}/\overline{T'U} = -\underline{t} \circ ((\underline{r} \times \underline{s}) \times \underline{s})/|\underline{s}| \cdot \underline{t} \circ (\underline{r} \times \underline{s})$. If, as is the case with our algorithm, the vector $\underline{s}$ is the same for all planes whose angles are to be compared, we may replace the comparison of cotangents with that of cotangents multiplied by $|\underline{s}|$. It is then straightforward to show that the computation of $|\underline{s}| \cdot \overline{SU}/\overline{T'U}$ requires four multiplications, four additions, and one division.

Once the construction of the triangulation $\mathcal{T}$ has been completed, i.e., the interleaving E_{AB} of E_A and E_B has been obtained, we must remove those portions of A and B which have become internal to CH(A,B). Concretely, this is done by constructing the data structure describing CH(A,B) from E_{AB} and from the data structures describing A and B. The data structure describing a spatial set C may be realized as a collection of lists $\{L(c)\}$, each list L(c) corresponding to a vertex c of C and giving the sequence of the edges incident on c, ordered according to the previously described criterion. By means of a vector of pointers, each list is accessible in fixed time.

We consider the lists of vertices in E_A and E_B. Let $E_A = a_{i_1}, a_{i_2}, \ldots, a_{i_r}$ and $E_B = b_{j_1}, b_{j_2}, \ldots, b_{j_s}$. Suppose we are currently updating the list $L(a_{i_k})$: typically, E_{AB} contains the substring $\gamma a_{i_k} b_{j_h} b_{j_{h+1}} \ldots b_{j_t} a_{i_{k+1}}$, where $b_{j_k} \ldots b_{j_t}$ is possibly empty and γ is either $a_{i_{k-1}}$ or $b_{i_{h-1}}$. Then we will remove from $L(a_{i_k})$ the edges comprised between $(a_{i_k}, a_{i_{k-1}})$ and $(a_{i_k}, a_{i_{k+1}})$, and insert the sequence $(a_{i_k}, \gamma)(a_{i_k}, b_{j_h}), \ldots, (a_{i_k}, b_{j_t})$: this effects the updating of $L(a_{i_k})$. In this manner we shall update the lists of all vertices in E_A and E_B: these are the only lists which need revision, since all other lists are either left unaltered or deleted altogether from the data structure.

The deletion of lists can be accomplished by a procedure very similar to topological sorting. We shall illustrate it by referring concretely to the polyhedron A. With each vertex a of A we associate a marker $\rho(a)$ which is initially set to 0. Next, for each vertex $a \in E_A$ we set $\rho(a) = 1$, and set $\rho(a') = 2$ for each vertex a' such that the edge $\overline{(a, a')}$ has been deleted when updating L(a); we also form a set V_A of all vertices which have received the marker $\rho = 2$ during the current step. At the subsequent step, for each $a \in V_A$, we shall set $\rho(a') = 2$ for each a' such that the edge $\overline{(a, a')}$ exists and $\rho(a') = 0$, and form V_A in the usual manner. The marking terminates at the step for which $V_A = \phi$, and the procedure is completed by deleting from the data structure each list $L(a)$ for which $\rho(a) = 2$.

The number of operations required by the updating procedure is proportional to the number of edges which are to be added when reconstructing the lists of the vertices in E_A and E_B, and to the number of edges that have to be inspected when deleting the lists. In the latter operation, each edge is inspected at most twice. Thus the total number of operation is proportional to the total number of edges of A and B, thereby yielding the conclusion that the number of operations is at most $O(p + q)$.

Therefore, since both the construction of the triangulation $\mathcal{T}$ and the deletion of obscured portions of A and B are procedures which require a number of operations at most linear in the number of vertices of A and B, this property holds for the merging algorithm as a whole, that is, $P_3(n) = O(n)$.

References

[1] E. N. Gilbert and H. Pollak, "Steiner minimal trees", SIAM Journal on Appl. Math., Vol. 16, pp. 1-29 (1968).

[2] J. Sklansky, "Measuring concavity on a rectangular mosaic", IEEE Transactions on Computers, Vol. C-21, pp. 1355-1364 (Dec. 1972).

[3] H. Freeman and R. Shapira, "Determining the minimum-area incasing rectangle for an arbitrary closed curve", to appear in the Comm. of the ACM (June 1975).

[4] R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set", Information Processing Letters, Vol. 1, pp. 132-133 (1972).

[5] R. A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane", Information Processing Letters, Vol. 2, pp. 18-21 (1973).

[6] B. Grünbaum, Convex Polytopes, Interscience, J. Wiley and Sons, New York, 1967.

[7] H. T. Kung, F. Luccio and F. P. Preparata, "On finding the maxima of a set of vectors", to appear in the Journal of the ACM.