JOURNAL OF COMPUTER AND SYSTEM SCIENCES **49**, 391–407 (1994)

Robust Gift Wrapping for the Three-Dimensional Convex Hull

KOKICHI SUGIHARA

Department of Mathematical Engineering and Information Physics, Faculty of Engineering, University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113, Japan

Received February 27, 1992; revised July 13, 1993

A conventional gift-wrapping algorithm for constructing the three-dimensional convex hull is revised into a numerically robust one. The proposed algorithm places the highest priority on the topological condition that the boundary of the convex hull should be isomorphic to a sphere, and uses numerical values as lower-priority information for choosing one among the combinatorially consistent branches. No matter how poor the arithmetic precision may be, the algorithm carries out its task and gives as the output a topologically consistent approximation to the true convex hull. © 1994 Academic Press, Inc.

1. INTRODUCTION

The convex hull of a finite number of points is one of the most fundamental concepts in computational geometry, and many efficient algorithms have been proposed for two dimensions [4, 6, 9, 14] and higher dimensions [1, 2, 3, 13, 15, 20]. However, these algorithms are designed in the implicit assumption that numerical computation can be carried out in exact arithmetic. Actual computation, on the other hand, is done in finite precision, and hence straightforward translation of these algorithms into computer language does not necessarily give practically valid computer programs; they may fail due to inconsistency caused by numerical errors.

In order to overcome this difficulty several approaches have been studied recently for the construction of the two-dimensional convex hull. Guibas *et al.* proposed a scheme for three-value logic, called epsilon geometry, and applied it for constructing an approximation to the convex hull in the plane [7]. Fortune used approximate geometric predicates in ordinary two-value logic and constructed a similar algorithm [5].

It seems, however, that their ideas cannot be extended to three dimensions directly because maintaining the topological consistency in three dimensions is nontrivial. Indeed, in two-dimensional space, any cyclic sequence of three or more points chosen from a finite set P of points can be an approximation of the convex hull of P in the sense that the cyclic sequence becomes the correct convex hull if the points in P are perturbed appropriately. In three dimensions, on the other hand, an arbitrary collection of triangles with vertices in P does not necessarily give the convex hull of P even if the points in P are perturbed.

0022-0000/94 $6.00

The convex hull of n points in three-dimensional space can be constructed in $O(n \log n)$ time by the divide-and-conquer algorithm, and this time complexity is known to be optimal [14]. Another famous algorithm is "gift wrapping," which runs in $O(kn)$ time, where k is the number of vertices on the boundary of the convex hull [2, 14, 20]. This algorithm is not optimal because k can be as large as n in the worst case, but it is still practically important because its time complexity is much smaller if k is small (e.g., linear in the case where k is a constant).

In this paper, we revise the conventional gift-wrapping algorithm into a numerically robust one. The idea here is the combinatorial abstraction proposed by Sugihara and Iri [18, 19]; the basic part of the algorithm is described in terms of combinatorial computation, and numerical values are used only to choose branches in the algorithm. The resultant algorithm is robust and topologically consistent; in any imprecise arithmetic the algorithm carries out its task to give some output, and the output can be the correct answer of the convex hull problem if the input points are perturbed appropriately.

After reviewing numerical problems in the conventional algorithm in Section 2, we extract the combinatorial structure of the algorithm in Section 3 and construct the new algorithm in Section 4. We also discuss the distance between the correct convex hull and the output of the proposed algorithm in Section 5.

2. Numerical Problems in the Conventional Method

Here we will review the conventional gift-wrapping method constructing the convex hull of a finite number of points in three-dimensional space, and we will see how numerical error causes the method to fail. For this purpose, let us start with the definition of the convex hull.

Let $\mathbf{R}$ be the set of real numbers, and let us denote by $\mathbf{R}^3$ the set of all the points in three-dimensional space to which a right-handed (x, y, z) coordinate system is fixed. Let $P = \{p_1, p_2, ..., p_n\}$ be a finite set of points in $\mathbf{R}^3$. The intersection of all the convex subsets of $\mathbf{R}^3$ containing P is called the *convex hull* of P and is denoted by $\mathrm{CH}(P)$. We assume that P contains at least three points and they are not collinear. P is said to be *coplanar* if all the points in P are on a common plane, and *noncoplanar* otherwise. If P is coplanar, $\mathrm{CH}(P)$ is a convex polygon, whereas if P is noncoplanar, $\mathrm{CH}(P)$ is a convex polyhedron.

From the algorithmic point of view, $\mathrm{CH}(P)$ can be obtained as the intersection of a finite number of half spaces in the following manner. For any four reals a, b, c, and d, where at least one of a, b, and c is nonzero, the set H of points defined by

$$H = \{(x, y, z) \mid ax + by + cz + d \geqslant 0\} \tag{2.1}$$

is called a *half space*, and the set of point ∂H defined by

$$\partial H = \{(x, y, z) \mid ax + by + cz + d = 0\} \tag{2.2}$$

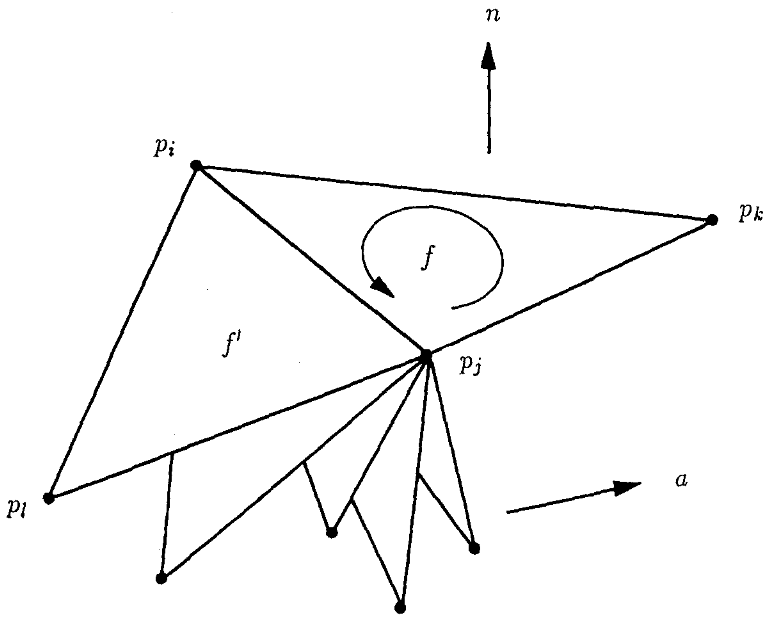


FIG. 1. One step of the gift wrapping.

is called the *boundary* of H. Half space H is said to be *critical with respect to P* if (i) $P \subseteq H$ and (ii) ∂H contains at least three noncollinear points in P. Then the next fact is well known.

Fact 1. Let P be noncoplanar. The convex hull of P is the intersection of all the critical half spaces with respect to P.

For any critical half space H with respect to P, $\partial H \cap \mathrm{CH}(P)$ is called a *face* of $\mathrm{CH}(P)$. Hence, there is a one-to-one correspondence between the critical half spaces with respect to P and the faces of $\mathrm{CH}(P)$. If no four points in P are coplanar (this situation is said to be *nondegenerate*), all the faces of $\mathrm{CH}(P)$ are triangular. If the intersection of two faces contains two or more points, the intersection is called the *edge* shared by the two faces. If the intersection of three or more faces is nonempty, the intersection is called the *vertex* shared with those faces. Every vertex of $\mathrm{CH}(P)$ is an element of P.

For two points p_i and p_j, let $\overline{p_i p_j}$ denote the undirected line segment connecting p_i to p_j, and let $\mathbf{p_i p_j}$ denote the directed line segment with the initial point p_i and the terminal point p_j. Let f be a triangular face of $\mathrm{CH}(P)$, and let p_i, p_j, and p_k be the three vertices of f. The face f has an orientation in the sense that $\mathrm{CH}(P)$ lies in one side of f and the other side is completely empty. We denote the face f by $\mathrm{face}(p_i, p_j, p_k)$ if the three directed line segments $\mathbf{p_i p_j}$, $\mathbf{p_j p_k}$, and $\mathbf{p_k p_i}$ surround f counterclockwise when we see it from the empty side and by $\mathrm{face}(p_i, p_k, p_j)$ if they surround f clockwise. Since the order is cyclic, we obtain

$$\mathrm{face}(p_i, p_j, p_k) = \mathrm{face}(p_j, p_k, p_i) = \mathrm{face}(p_k, p_i, p_j).$$

In what follows, we mean by a face an oriented face having one of the two possible orientations in the above sense. The three directed line segments $\mathbf{p_i p_j}$, $\mathbf{p_j p_k}$, and $\mathbf{p_k p_i}$ are called *directed edges induced by* face(p_i, p_j, p_k).

In what follows, we assume that P is nondegenerate so that no four points are coplanar (for our purpose it is enough to consider only the nondegenerate case; we will discuss this in Section 3). Fact 1 directly gives us an algorithm for constructing the convex hull. Starting with one face, we find adjacent faces step by step, just as we wrap P using an ideally elastic membrane.

One step of wrapping is depicted in Fig. 1. Let $f = \text{face}(p_i, p_j, p_k)$ be a face that has already been found. We now want to find the other face that shares the edge $\mathbf{p_i p_j}$. Let n be the unit vector crossing f orthogonally from inside CH(P) toward the outside, as shown in Fig. 1. Let $\mathbf{a}$ be the unit vector defined by

$$\mathbf{a} = \frac{\mathbf{n} \times \mathbf{p_i p_j}}{|\mathbf{n} \times \mathbf{p_i p_j}|}. \tag{2.3}$$

The vector $\mathbf{a}$ is perpendicular to both $\mathbf{n}$ and $\mathbf{p_i p_j}$; the three vectors $\mathbf{n}$, $\mathbf{p_i p_j}$, and $\mathbf{a}$ form the right-handed mutually orthogonal system just as the thumb, the index finger, and the middle finger of the right hand can form. Let P' be the set of points in P that are not on f. We find the point, say p_l, in P' that attains the minimum of

$$g(f, p_l) = \frac{\mathbf{a} \cdot \mathbf{p_i p_l}}{|\mathbf{a} \cdot \mathbf{p_i p_l}|} \tag{2.4}$$

over all points in P'. The value $g(f, p_l)$ being minimum means that the angle θ_l between the face f and the triangle formed by p_i, p_j, and p_l is maximum, where the angle θ_l is measured through the interior of the convex hull (note that all the points in P' are below the plane containing f, and $g(f, p_l) = \cos\theta_l$). The face $f' = \text{face}(p_j, p_i, p_l)$ is the face that we want to find.

Let E_1 and E_2 be two sets of directed edges. We define $E_1 \oplus E_2$ by

$$E_1 \oplus E_2 = \{\mathbf{p_i p_j} \mid \mathbf{p_i p_j} \in E_1 \cup E_2, \mathbf{p_j p_i} \notin E_1 \cup E_2\}. \tag{2.5}$$

That is, directed edge $\mathbf{p_i p_j}$ is an element of $E_1 \oplus E_2$ if and only if $\mathbf{p_i p_j}$ is contained in $E_1 \cup E_2$ but the reversal $\mathbf{p_j p_i}$ is not contained in either E_1 or E_2. Hence, for example, if $E_1 = \{\mathbf{p_1 p_2}, \mathbf{p_2 p_3}, \mathbf{p_3 p_4}, \mathbf{p_4 p_1}\}$ and $E_2 = \{\mathbf{p_2 p_1}, \mathbf{p_1 p_5}, \mathbf{p_5 p_2}\}$ as shown in Fig. 2a, we obtain $E_1 \oplus E_2 = \{\mathbf{p_2 p_3}, \mathbf{p_3 p_4}, \mathbf{p_4 p_1}, \mathbf{p_1 p_5}, \mathbf{p_5 p_2}\}$ as shown in Fig. 2b. Thus, for two sets F_1 and F_2 of faces, if E_1 and E_2 are the sets of the boundary edges of the surfaces composed of the faces in F_1 and F_2, respectively, $E_1 \oplus E_2$ is the set of the boundary edges of the surface composed of the faces in $F_1 \cup F_2$.

The gift-wrapping method is described by the following algorithm, where Q is a queue containing faces whose boundary edges have to be checked and E is the storage containing the edges, one of whose faces has not yet been found.

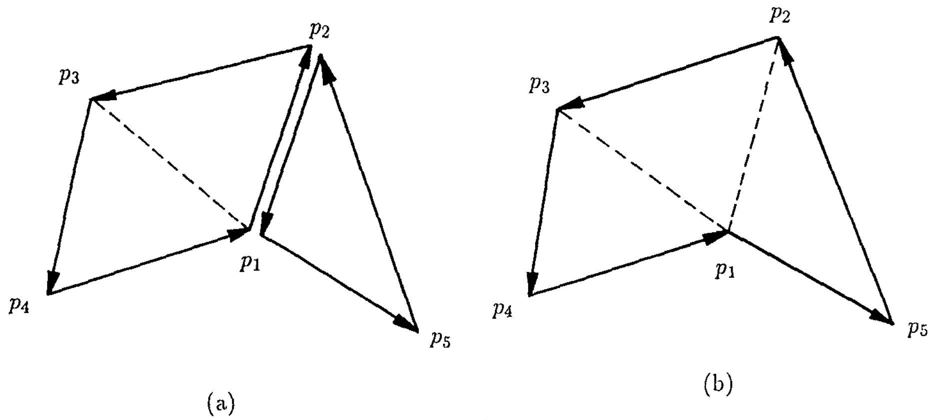


FIG. 2. Merging two surface boundaries: (a) before merging; (b) after merging.

ALGORITHM 1 (Conventional gift wrapping).
Input: set $P = \{p_1, p_2, ..., p_n\}$ of points in $\mathbf{R}^3$.
Output: set F of all the faces on the boundary of CH(P).

Procedure:
1. Find one face f_0 on CH(P), $Q \leftarrow \{f_0\}$ and $F \leftarrow \varnothing$.
2. Let E be the set of all the directed edges on the boundary of f_0.
3. While Q is not empty do
   begin
   3.1. choose and delete face f from Q,
   3.2. $A \leftarrow$ the set of all the directed edges on the boundary of f,
   3.3. for each edge $e = \mathbf{p}_i\mathbf{p}_j \in A \cap E$ do
      begin
      3.3.1. find the point p_l that minimizes $g(f, p_l)$, and $f' \leftarrow$ face(p_j, p_i, p_l),
      3.3.2. $B \leftarrow$ the set of all the directed edges on the boundary of f',
      3.3.3. $E \leftarrow E \oplus B$,
      3.3.4. add f' to Q
      end
   3.4. add f to F.
   end
4. Report F and stop.

This algorithm constructs the convex hull of P correctly if no error takes place in the course of numerical computation [14]. If numerical error takes place, on the other hand, the validity of the algorithm is not guaranteed.

An example of the situation in which the algorithm fails is shown in Fig. 3. We denote by triangle(p_i, p_j, p_k) the triangle formed by the three vertices p_i, p_j, and p_k (when we say triangle(p_i, p_j, p_k), we do not care about the orientation). Let

$P=\{p_1, p_2, \ldots, p_5\}$ such that p_1, p_2, p_3, and p_4 form a tetrahedron and p_5 is almost on triangle(p_1, p_3, p_4). This implies that we may not be able to correctly judge the relative position of p_5 with respect to triangle(p_1, p_3, p_4) due to numerical error. The result of numerical judgement can be any one of the three: (i) p_5 and p_2 are in mutually opposite sides of triangle(p_1, p_3, p_4); (ii) p_5 and p_2 are in the same side of triangle(p_1, p_3, p_4); (iii) p_5 is on triangle(p_1, p_3, p_4). Suppose that in Step 1 of Algorithm 1 we find

$$f_1 = \text{face}(p_1, p_3, p_2)$$

as the initial face and put it in $Q: Q=\{f_1\}$. In Step 2, E is set as $E=\{\mathbf{p}_1\mathbf{p}_3, \mathbf{p}_3\mathbf{p}_2, \mathbf{p}_2\mathbf{p}_1\}$. In Step 3.1, $f=f_1$ is chosen. In the first repetition of Step 3.3, we choose edge $\mathbf{p}_2\mathbf{p}_1$ and find the adjacent face $f_2=\text{face}(p_1, p_2, p_4)$; hence E is changed to $\{\mathbf{p}_1\mathbf{p}_3, \mathbf{p}_3\mathbf{p}_2, \mathbf{p}_2\mathbf{p}_4, \mathbf{p}_4\mathbf{p}_1\}$. In the second repetition of Step 3.3, we choose edge $\mathbf{p}_3\mathbf{p}_2$ and find the adjacent face $f_3=\text{face}(p_2, p_3, p_4)$; consequently E is changed to $\{\mathbf{p}_1\mathbf{p}_3, \mathbf{p}_3\mathbf{p}_4, \mathbf{p}_4\mathbf{p}_1\}$. In the third repetition of Step 3.3, we choose edge $\mathbf{p}_1\mathbf{p}_3$ and search for the adjacent face. Suppose that numerical computation tells us that the angle between f_1 and triangle(p_1, p_3, p_5) is greater than the angle between f_1 and triangle(p_1, p_3, p_4), and consequently face(p_1, p_5, p_3) is recognized as the adjacent face. Then, E is changed to $\{\mathbf{p}_1\mathbf{p}_5, \mathbf{p}_5\mathbf{p}_3, \mathbf{p}_3\mathbf{p}_4, \mathbf{p}_4\mathbf{p}_1\}$. Now in the second repetition of Step 3.1, $f=f_2$ is chosen and in Step 3.3, we choose edge $\mathbf{p}_4\mathbf{p}_1$ and search for the adjacent face. Suppose that numerical computation judges that the angle between f_2 and triangle(p_1, p_3, p_4) is greater than the angle between f_2 and triangle(p_1, p_4, p_5). This judgement is not consistent with the previous judgement that the angle between f_1 and triangle(p_1, p_3, p_5) is greater than the angle between f_1 and triangle(p_1, p_3, p_4). However, this can happen because of numerical error in the computation of (2.4). Then, face(p_1, p_4, p_3) is recognized as the adjacent face. At this stage, E is changed to $\{\mathbf{p}_1\mathbf{p}_5, \mathbf{p}_5\mathbf{p}_3, \mathbf{p}_3\mathbf{p}_1\}$. Now we have a topological

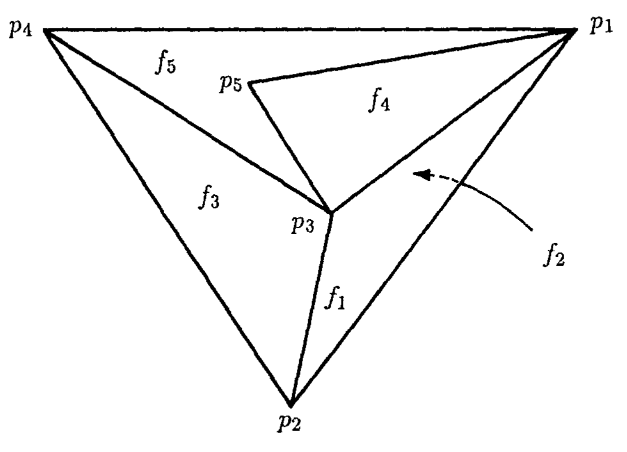


FIG. 3. Topological inconsistency due to numerical errors.

inconsistency because the undirected edge $\overline{p_1 p_3}$ is shared by the three faces f_1, f_4, and f_5, which should not happen on the boundary of CH(P).

Some of the readers might think that inconsistency could be avoided if we are careful in such a way that we do not generate an edge shared by three faces. However, the problem is not so simple. Another example of inconsistency is shown in Fig. 4, where P consists of six points $p_1, p_2, \ldots, p_6$ such that $p_2, p_3, \ldots, p_6$ are almost coplanar, forming a convex pentagon, as in (a). Suppose that we start

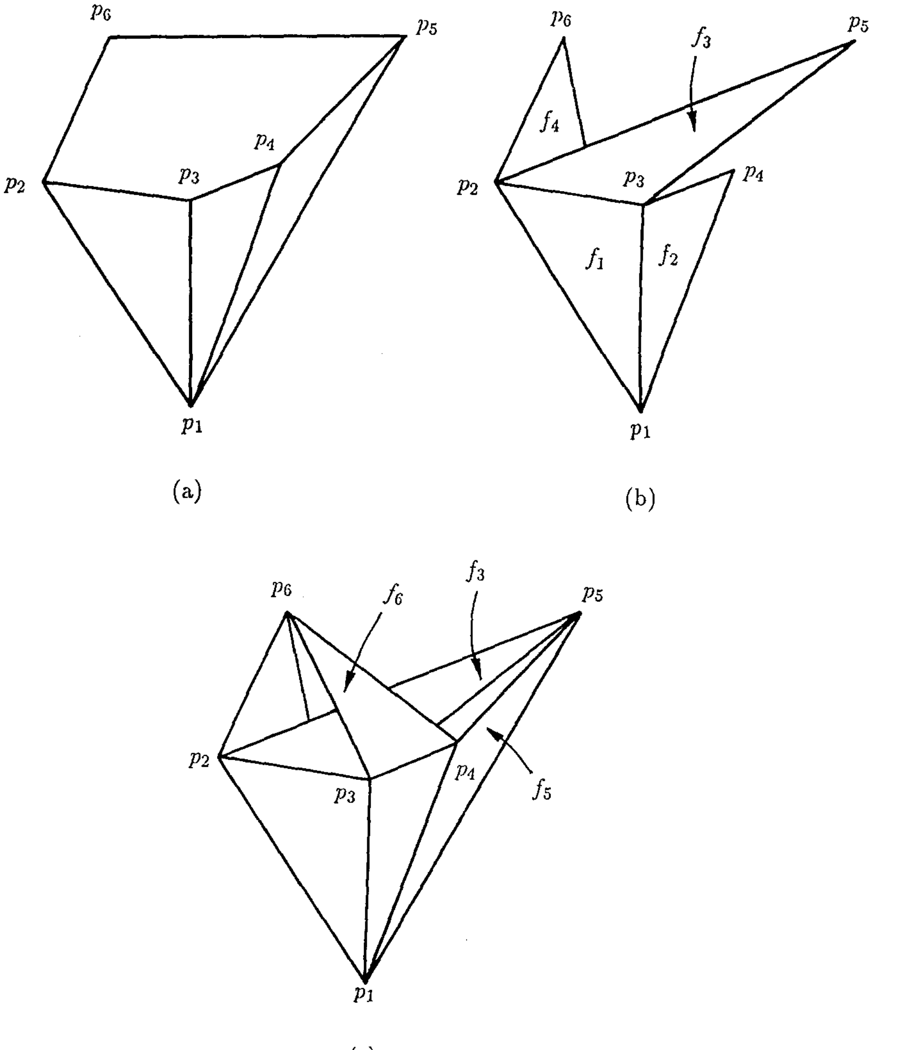


FIG. 4. Another example of topological inconsistency: (a) correct convex hull; (b) midway of the gift wrapping with numerical errors; (c) inconsistency.

with the initial face $f_1 = \text{face}(p_1, p_3, p_2)$, in the first repetition of Steps 3.1–3.3 we find three adjacent faces $f_2 = \text{face}(p_1, p_4, p_3)$, $f_3 = \text{face}(p_2, p_3, p_5)$, and $f_4 = \text{face}(p_1, p_2, p_6)$, as shown in (b), and that in the second repetition we choose f_2 and find two adjacent faces $f_5 = \text{face}(p_1, p_5, p_4)$ and $f_6 = \text{face}(p_3, p_4, p_6)$, as shown in (c). This can happen due to numerical error because $p_2, p_3, ..., p_6$ are almost coplanar. The set of faces generated so far has the property that no three faces share a common (undirected) edge. However, we cannot augment the set of faces in such a way that the resultant set of faces forms a surface homeomorphic to a sphere. Thus, we have inconsistency.

3. Combinatorial Abstraction of Gift Wrapping

In this and the next sections we construct a numerically robust gift-wrapping method, i.e., a method that does not fail even if numerical error takes place. For this purpose we employ the combinatorial abstraction approach proposed by Sugihara and Iri [18, 19], in which the basic part of the algorithm is described in terms of combinatorial computation and numerical computation is used only to select an appropriate branch of the processing. By this approach we can construct an algorithm that is free from topological inconsistency in any finite-precision arithmetic.

In imprecise arithmetic it is impossible to construct the convex hull $\text{CH}(P)$ always correctly. Hence, we change our goal; instead of constructing the true convex hull, we aim at constructing an approximation to the convex hull. In this context we can assume that P is nondegenerate so that any four points in P are noncoplanar. This assumption does not lose generality because of the following reason. Suppose that P is degenerate. We perturb the points in P slightly to obtain a nondegenerate point set, say P', and construct an approximation to $\text{CH}(P')$. Then, we can expect that this approximation is also an approximation to $\text{CH}(P)$, because the perturbation is small. Moreover, it is impossible to discern degeneracy in imprecise arithmetic. Hence, it is not at all a restriction to assume that P is nondegenerate. Consequently, we consider only the case where all the faces of $\text{CH}(P)$ are triangular.

Suppose that P has four or more points. Since P is nondegenerate, $\text{CH}(P)$ has a nonzero volume and hence the boundary $\partial\text{CH}(P)$ is isomorphic to a sphere, implying that the graph composed of the edges and the vertices of $\text{CH}(P)$ is planar. This observation demands that the gift-wrapping procedure should be carried out in such a way that the final graph is planar. Since a subgraph of a planar graph is planar, the gift-wrapping procedure should also be carried out in such a way that the graph obtained at every step is planar. However, this demand is not sufficient; for example, the graph composed of the vertices and the edges of the triangles in the structure shown in Fig. 3 is planar but it is topologically inconsistent. In order to detect inconsistency without backtracking, we need to consider not only the vertices and edges but also the faces.

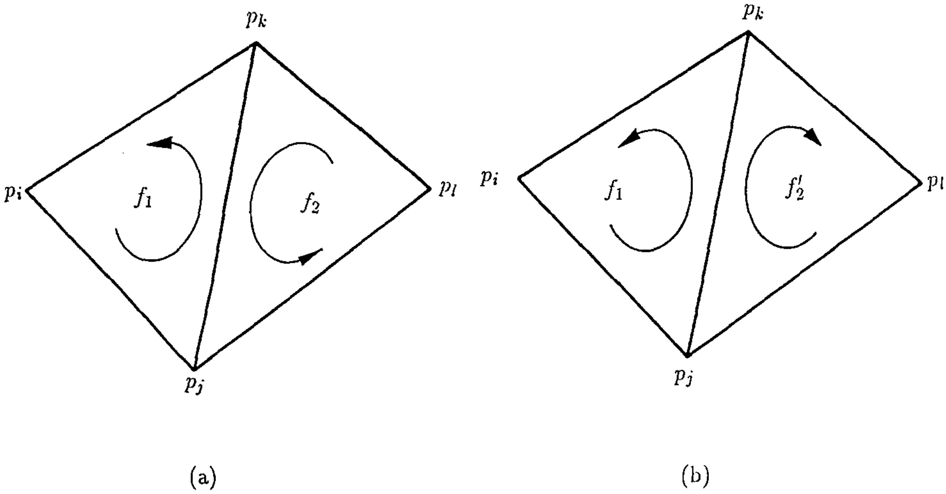


FIG. 5. Consistency of the face orientations: (a) two faces having consistent orientations; (b) two faces having inconsistent orientations.

Let f_1 and f_2 be two faces sharing two vertices p_j and p_k. We say that f_1 and f_2 *have consistent orientations* if $\mathbf{p_j p_k}$ is a directed edge induced by one face and $\mathbf{p_k p_j}$ is a directed edge induced by the other. Hence, $f_1 = \text{face}(p_i, p_j, p_k)$ and $f_2 = \text{face}(p_l, p_k, p_j)$ have consistent orientations (Fig. 5a), whereas $f_1 = \text{face}(p_i, p_j, p_k)$ and $f_2' = \text{face}(p_l, p_j, p_k)$ do not (Fig. 5b). Note that $\text{face}(p_i, p_j, p_k)$ and $\text{face}(p_i, p_k, p_j)$ (they have the same vertex set) have consistent orientations. A set F of faces is said to have *consistent orientations* if for any $f_i, f_j \in F$, f_i and f_j have consistent orientations or they share no edge. If F has consistent orientations, any undirected edge can be shared by at most two faces in F.

If P has four or more points, the set of the faces of $\text{CH}(P)$ has consistent orientations. If P has exactly three points, say $P = \{p_1, p_2, p_3\}$, then we consider $\{\text{face}(p_1, p_2, p_3), \text{face}(p_2, p_1, p_3)\}$ as the set of the faces of $\text{CH}(P)$. Hence for any point set P having three or more points, the set of the faces of $\text{CH}(P)$ has consistent orientations.

The vertices and the edges of $\text{CH}(P)$ form a planar graph. Let us draw this graph in such a way that the edges do not intersect except at their endpoints. An example of such a drawing is shown in Fig. 6a. There are exactly three outermost edges, forming the largest triangle, and the interior of this triangle is decomposed into mutually nonoverlapping triangles. The outermost triangle and the nonoverlapping triangles correspond to the faces of $\text{CH}(P)$. As the outermost triangle we choose the initial face for gift wrapping, i.e., the face found in Step 1 of Algorithm 1. We consider that this face is drawn from a viewpoint lying in the same side as the convex hull; hence the orientation of this face is clockwise as shown in the figure. The gift-wrapping algorithm starts with this outermost triangle and add other triangles in the interior. These triangles correspond to the faces seen from the empty side, and

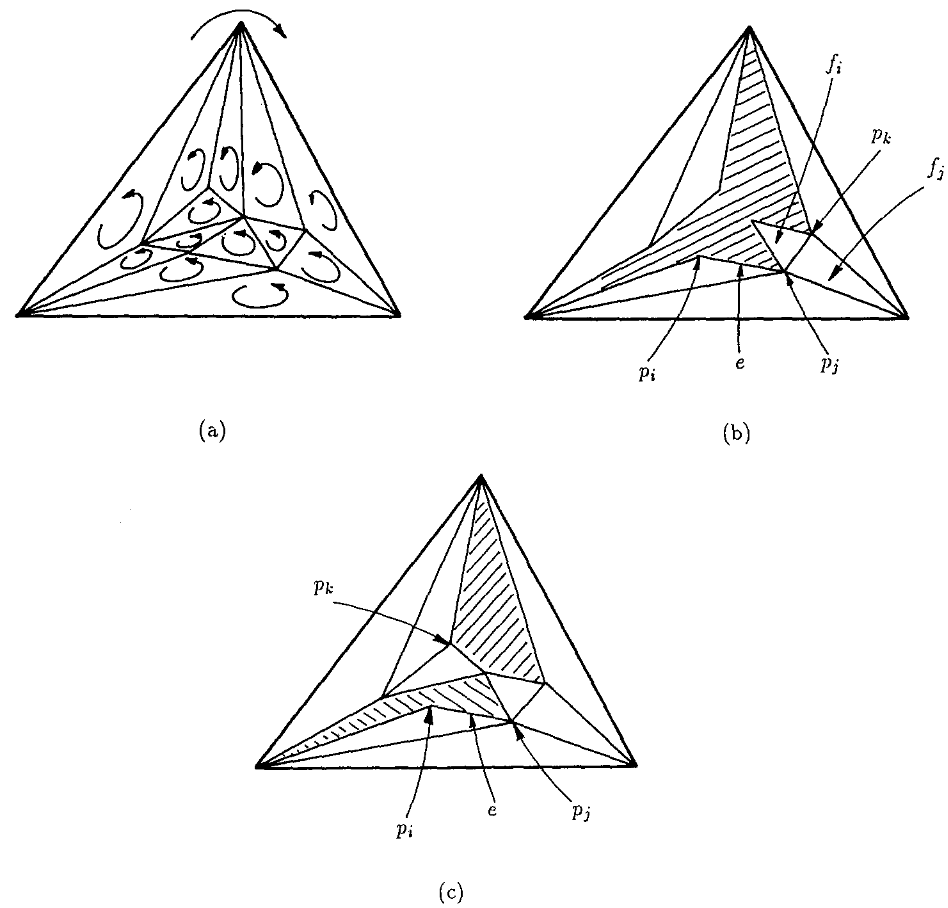


FIG. 6. Representation of the topological structure of the convex hull: (a) convex hull; (b) partial structure of the convex hull having one unfilled region; (c) partial structure of the convex hull having two unfilled regions.

consequently their orientations are all counterclockwise. Typical examples of the structures obtained by gift wrapping are shown Figs. 6b and c, where the shaded regions represent the areas that have not yet been filled with the faces. Let us call these regions *unfilled regions*. In earlier stages of gift wrapping, there is only one unfilled region as shown in b, but in general this region is partitioned into two or more regions as shown in c. The edges on the boundary of the unfilled regions are the edges at which the next face is searched for.

Suppose that at some of the gift wrapping we have found k faces $f_1, f_2, ..., f_k$. Let F be the set of these faces: $F = \{f_1, f_2, ..., f_k\}$. Let $E(f_i)$ be the set of directed edges that are on the boundary of f_i and let us define

$$E(F) = E(f_1) \oplus E(f_2) \oplus \cdots \oplus E(f_k).$$

The edges in $E(F)$ form the boundaries of the unfilled regions. For each $e \in E(F)$, let $S(e)$ be the set of vertices on the boundary of the unfilled region that contains e. Let, furthermore, $I(F)$ be the set of points in P that are not the vertices of any face in $E(F)$. Elements of $I(F)$ are called *isolated points.*

Suppose that Algorithm 1 chooses edge $e = \mathbf{p_i p_j}$ in $E(F)$ in Step 3.3 and finds the new face, say f', incident to e. Let p_k be the third vertex of f', that is, $f' = \text{face}(p_j, p_i, p_k)$. The new face f' should satisfy the next two conditions:

(C1) $p_k \in S(e) \cup I(F)$.

(C2) Each of undirected edges $\overline{p_i p_k}$ and $\overline{p_j p_k}$ is incident to at most one face in F.

Condition (C1) says that the third vertex p_k of the new face should either be on the boundary of the same unfilled region as e or an isolated vertex. This condition should be satisfied because otherwise the surface that is composed of the faces constructed by gift wrapping is not homeomorphic to the sphere. For example, if we search for the new face incident to edge e in Fig. 6c, the vertex p_k cannot be the third vertex of the new face because e and p_k belong to different unfilled regions. The situation shown in Fig. 4c also violates (C1).

Condition (C2) should be satisfied because otherwise three faces share a common edge, which cannot happen for the surface that is homeomorphic to the sphere. For example, if we search for the new face that is incident to e in Fig. 6b, p_k in this figure cannot be the third vertex of the new face, because the edge $\overline{p_j p_k}$ is already shared by two faces f_i and f_j. The inconsistency shown in Fig. 3 can also be avoided by (C2).

4. Robust Gift-Wrapping Algorithm

Using conditions (C1) and (C2), we can revise Algorithm 1 into the next one.

Algorithm 2 (Robust gift wrapping).
Input: set $P = \{p_1, p_2, ..., p_n\}$ of points in $\mathbf{R}^3$ $(n \geqslant 3)$.
Output: set F of faces that approximates the boundary of CH(P).

Procedure: the same as the procedure in Algorithm 1 except that Step 3.3.1 is replaced by the next one.

3.3.1. Among all the points in $S(e) \cup I(e)$, find the one, say p_l, that minimizes $g(f, p_l)$, and $f' \leftarrow \text{face}(p_j, p_i, p_l)$.

We consider the behavior of Algorithm 2 in the world where the numerical error takes place. By numerical computation we mean computation with real numbers (typically represented by floating-point numbers). Here we assume that the error takes place in the numerical computation, and, moreover, we assume that the amount of error cannot be bounded. The latter assumption might seem too pessimistic because usually the errors are small. However, our approach is strong enough to be able to construct a robust algorithm even in this pessimistic assumption. Combinatorial computation, on the other hand, is assumed to be carried out without any error.

A convex-hull algorithm is said to be *robust* if in any imprecise arithmetic it does not come across topological inconsistency and, hence, it carries out the task, ending up with some output. A convex-hull algorithm is said to be *topologically consistent* if it is robust and its output is a correct set of faces of the convex hull of some perturbation of the input point set P. A convex-hull algorithm is said to be *stable* if it is topologically consistent and if the maximum displacement to perturb the input points to make the output correct is bounded by a constant depending on the precision. We can see that Algorithm 2 is robust and topologically consistent but not stable.

THEOREM 1. *Algorithm 2 is robust.*

Proof. Algorithm 2 employs numerical computation only in Step 1 and Step 3.3.1. Due to numerical error the algorithm may not find the correct face corresponding to the true convex hull. However, some face is obtained in Step 1 and Step 3.3.1. There are only a finitely many ordered triples of points in P, and the same triple is never obtained twice or more in Step 3.3.1. Therefore, Algorithm 2 terminates in finitely many steps, which implies that Algorithm 2 is robust. ∎

Graph G is said to be *triply connected* if the deletion of any two vertices and the edges incident to them from G does not make the remaining graph disconnected. The next theorem is helpful to prove the topological consistency of Algorithm 2. See Steinitz [16] (or Lyusternik [10]) for the proof of this theorem.

THEOREM 2 (Steinitz' theorem [16]). *For any triply connected planar graph G with four or more vertices, there exists a convex polyhedron in three-dimensional space such that the graph composed of the vertices and the edges of the polyhedron is isomorphic to G.*

THEOREM 3. *Algorithm 2 is topologically consistent.*

Proof. Case 1. Suppose that the output of Algorithm 2 consists of only two faces, say, $F = \{\text{face}(p_i, p_j, p_k), \text{face}(p_i, p_k, p_j)\}$. Let us move all the other points in P to any places in the triangle formed by the vertices p_i, p_j, and p_k and let the resultant set of points be P'. Then, F is the set of the faces of $\text{CH}(P')$. Thus, we obtain the perturbation P' of P such that the output is exactly the face set of $\text{CH}(P')$.

Case 2. Suppose that the output F of Algorithm 2 consists of three or more faces. In this case the number of vertices in some faces in F is at least four (and, moreover, the number of faces in F is also at least four). The face obtained at any repetition of Step 3.3.1 satisfies the conditions (C1) and (C2), and consequently the set F of faces always has a consistent orientation and the graph G composed of the vertices and the edges of the faces in F is always planar. In particular, at the end of the algorithm, the graph G is planar and every edge of G is incident to exactly two faces, because if there is an undirected edge e incident to only one face f, f is put in Q in Step 1 or in Step 3.3.4 and some other face incident to e is found when f is deleted from Q. Thus, the graph G at the end of the algorithm is a triangular planar graph which is triply connected. Let X be a convex polyhedron whose vertex-edge graph is isomorphic to G; the existence of such a convex polyhedron is guaranteed by Steinitz' theorem. Since G is triangular, all the faces of X are triangles. Let us move the vertices of G to the locations of the corresponding vertices of X, let us move the remaining points in P to the interior of X, and let the resultant set of points be P'. Thus, we obtain the perturbation P' of P such that the output of Algorithm 2 is the set of faces of $\mathrm{CH}(P')$, which implies that Algorithm 2 is topologically consistent. ▮

As we have seen, Algorithm 2 is robust and topologically consistent, but it is not stable as seen in the next example.

As shown in Fig. 7, suppose that the point set P is almost coplanar, and, hence, $\mathrm{CH}(P)$ is almost a convex polygon. Assume that Step 1 of Algorithm 2 finds the initial face $f = \mathrm{face}(p_i, p_j, p_k)$, such that p_i and p_j are on the boundary of the convex polygon but that p_k is in its interior, and assume that in the first execution of Step 3.3 the edge $\mathbf{p_i p_j}$ is chosen. Since the points are almost coplanar, any point is likely to be chosen as the third vertex of the new face f'. Assume that p_k is chosen as the third vertex. Note that this choice does not violate condition (C1) or (C2). Then, all the edges are shared by two faces, and consequently Algorithm 2 terminates with the output $F = \{\mathrm{face}(p_i, p_j, p_k), \mathrm{face}(p_i, p_k, p_j)\}$. In order to make F the correct face set of a convex hull, we need to perturb P in such a way that the triangle with the vertices p_i, p_j, and p_k contains all the other points. However, there exists point set P that requires arbitrarily large perturbation, because the other points can be aribtrarily far from the triangle. Thus, Algorithm 2 is not stable.

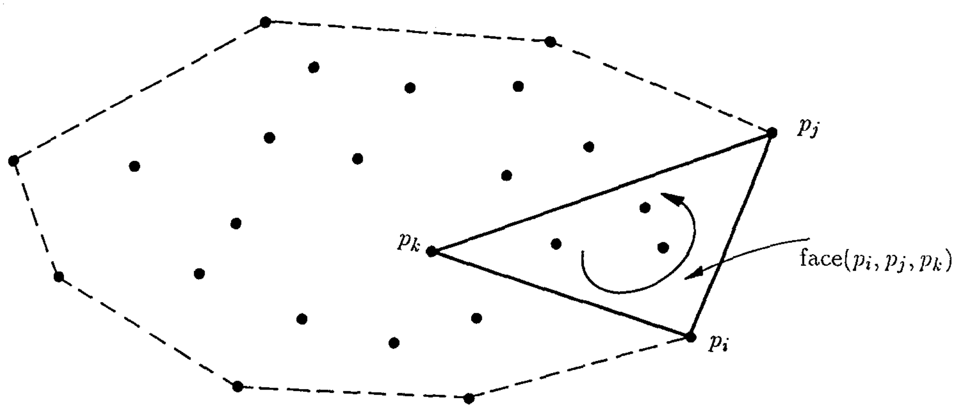


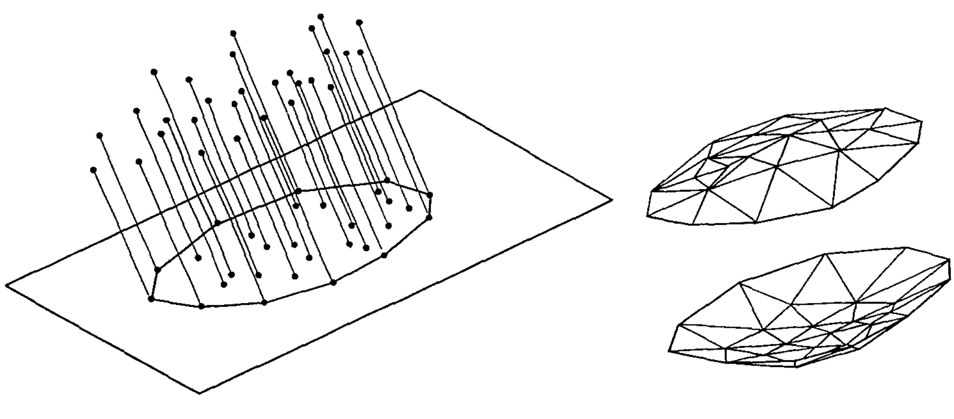
FIG. 7. Gift wrapping for almost planar points.

THEOREM 4. *The time complexity of Algorithm 2 is of $O(kn)$, where k is the number of faces in the output F.*

Proof. The only difference of Algorithm 2 from the conventional algorithm (Algorithm 1) is that conditions (C1) and (C2) are checked in Step 3.3.1. This check is done in $O(n)$ time in the following manner. We create list IL of all the isolated points. For the sth unfilled region ($s = 1, 2, \ldots$), we create cyclic list CL[s] of the vertices and the directed edges on its boundary, and to each edge e in CL[s] we assign label $\alpha[e]$ representing the unfilled region number to which e belongs. We create another list EL storing the edges ever stored in E in Step 2 or Step 3.3.3. When edge $e = \mathbf{p}_i\mathbf{p}_j$ is chosen in Step 3.3, all the elements in $S(e)$ and in $I(e)$ can be retrieved by CL[$\alpha[e]$] and IL, and for each $p_l \in S(e) \cup I(e)$, condition (C2) can be checked by the list EL. Since the graph composed of the vertices in P and the boundary edges of the faces constructed in Algorithm 2 is planar, the number of the faces and that of the edges are of $O(n)$, and consequently the above check is done in $O(n)$ time. Also the modification of IL, CL[s], $\alpha[e]$, EL at each repetition of Step 3.3.1 can be done in $O(n)$ time. Step 3.3.1 of the conventional algorithm also requires $O(n)$ time, and, hence, the revision from Algorithm 1 to Algorithm 2 does not increase the time complexity. Therefore, as proved in [14], the algorithm runs in $O(nk)$ time. ▮

5. TOWARD A STABLE GIFT WRAPPING

It seems that the unstableness of Algorithm 2 is mainly due to the case where P is almost coplanar. This problem might be solved by the following heuristic modification.



(a) (b)

FIG. 8. Heuristic for avoiding unstableness: (a) two-dimensional convex hull for the projected points; (b) upper and lower convex hulls of the original points.

Modification of Algorithm 2. At the beginning of the algorithm we judge whether P is almost coplanar. If it is not, Algorithm 2 is done. If P is almost coplanar, then: (i) we project P onto a plane that is parallel to the best fitting plane to P, as shown in Fig. 8a; (ii) construct the two-dimensional convex hull of the projected points; and (iii) use the boundary of this convex hull twice as the initial boundary of the unfilled region for three-dimensional gift wrapping in the upper side and in the lower side, as shown in Fig. 8b.

Note that the initial face constructed in Step 1 of Algorithm 2 is used to obtain the initial boundary of the unfilled region. Hence, it need not be triangular, and, moreover, it need not be planar. What we need is a cyclic list of (not necessarily planar) points forming the boundary of the unfilled region. Therefore, the boundary of the convex hull of the projected points, when reversely mapped to the original three-dimensional space, can be used as the initial boundary from which we start gift wrapping. Thus, the above modification of Algorithm 2 is almost straightforward.

However, even if we employ this modification, it is not easy to guarantee the stableness mathematically. This is because in general it requires complicated analysis to establish the quantitative relations between the triply connected planar graph and the convex polyhedron [8], although the qualitative relation is established by Steinitz' theorem.

For example, let $P=\{p_1, p_2, ..., p_n\}$ be the set of the vertices of a two-dimensional convex polygon having the edge set $E_0=\{\overline{p_1p_2}, \overline{p_2p_3}, ..., \overline{p_{n-1}p_n}, \overline{p_np_1}\}$, as shown in Fig. 9, where $n=8$. Let E_1 and E_2 be mutually disjoint sets of diagonals each giving a triangular partition of the polygon; in Fig. 9 the diagonals belonging to one set are represented by solid lines and those belonging to the other, by broken lines. Now the edge set $E=E_0 \cup E_1 \cup E_2$ gives a planar triangular graph, so that

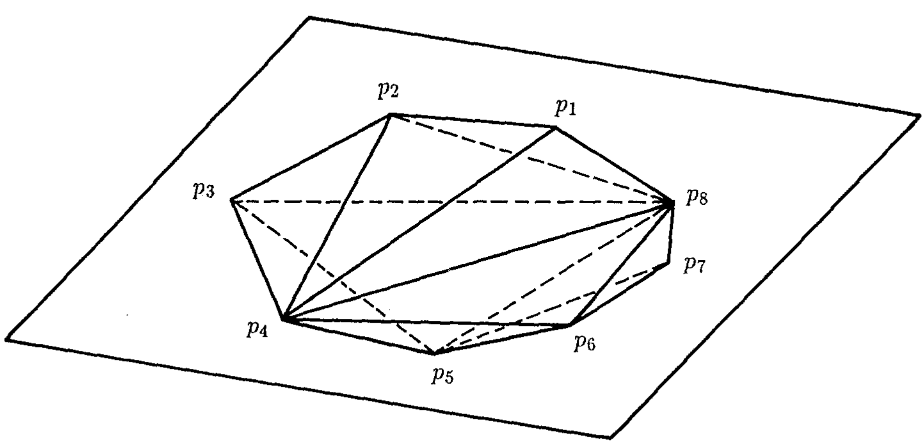


FIG. 9. Two sets of diagonal edges, each giving a triangulation of the convex polygon.

there exists a convex polyhedron whose vertex–edge graph is isomorphic to this graph.

However, it is not so obvious that the convex polyhedron can be realized by small perturbation of the points. Suppose that at the beginning all the eight points $p_1, p_2, ..., p_8$ are coplanar as shown in Fig. 9. If we pull p_4 upward slightly and p_7 downward sufficiently, then we obtain the upper convex hull of the eight points as indicated by the solid edges in Fig. 9. However, the lower convex hull of these points does not necessarily coincide with the triangulation indicated by the broken edges. Moreover, it is not trivial to see whether we can move the points to make the lower convex hull into the same structure as indicated by the broken edges while keeping the upper convex hull unchanged.

Guibas pointed out that it is open to judge whether the polyhedron can be realized by (not necessarily small) perturbation of the points in the direction perpendicular to the plane containing P [12]. Note that Algorithm 2, as well as its modification, can generate the output corresponding to E. Hence, in order to argue about the stableness of the algorithm, we have to establish the quantitative version of Steinitz' theorem. This is a problem for the future.

6. Concluding Remarks

We have proposed a numerically robust and topologically consistent version of the gift-wrapping algorithm for constructing the three-dimensional convex hull. On the basis of Steinitz' theorem, we add to the conventional gift-wrapping algorithm some combinatorial tests for avoiding topological inconsistency. No matter how poor the precision in computation may be, the purposed algorithm carries out the task and gives an output that is at least topologically consistent. Moreover, the new algorithm has the same time complexity as the conventional one; the combinatorial tests do not increase the time complexity. On the other hand, we have not yet succeeded in proving the stableness of the algorithm, which is a problem for the future.

Another possible direction to discuss the stableness is to enrich the object world from convex polyhedra to "nearly" convex polyhedra, just as the line arrangement is enriched to the pseudo-line arrangement [11]. Consider the example in Fig. 9 again. Indeed, if we can ignore the convexity, we can generate all the triangular faces for any perturbation of P. However, it seems difficult to make sure that such triangular faces do not intersect one another. Thus, the problem still seems nontrivial. More detailed discussion on the stableness in this sense will be presented in another paper.

The approach taken in this paper, i.e., the combinatorial abstraction, can be applied for constructing numerically robust algorithms for many other geometric problems. For example, other types of algorithms for the three-dimensional convex hull, such as a beneath-beyond method and a divide-and-conquer method, can also be modified into robust ones [17].

ACKNOWLEDGMENTS

The author is thankful to the anonymous referee, whose valuable comments on the earlier version of the manuscript made the paper much more readable. The work is supported by the Grant in Aid for Scientific Research of the Ministry of Education, Science, and Culture of Japan (Grant Nos. 04452191 and 05555027).

REFERENCES

1. C. E. BUCKLEY, A divide-and-conquer algorithm for computing 4-dimensional convex hull, *in* "International Workshop on Computational Geometry, Wurzburg, March 1988," Lecture Notes in Computer Science, Vol. 333, pp. 113–135, Springer-Verlag, Berlin, 1988.
2. D. R. CHAND AND S. S. KAPUR, An algorithm for convex polytopes, *J. Assoc. Comput. Mach.* **17** (1970), 78–86.
3. B. M. CHAZELLE, "An Optimal Convex Hull Algorithm for Point Sets in Any Dimension," Technical Report CS-TR-336-91, Princeton University, 1991.
4. W. EDDY, A new convex hull algorithm for planar sets, *ACM Trans. Math. Software* **3** (1977), 398–403.
5. S. FORTUNE, Stable maintenance of point-set triangulations in two dimensions, *in* "Proceedings, 30th Annual Symposium on Foundations of Computer Science," pp. 494–499, IEEE Comput. Soc. Press, Los Alamitos, CA, 1989.
6. R. L. GRAHAM, An efficient algorithm for determining the convex hull of a finite planar set, *Inform. Process. Lett.* **1** (1972), 132–133.
7. L. GUIBAS, D. SALESIN, AND J. STOLFI, Epsilon geometry—Building robust algorithms from imprecise calculations, *in* "Proceedings, 5th Annual Symposium on Computational Geometry," pp. 208–217, ACM Press, New York, 1989.
8. J E. HOPCROFT AND P. J. KAHN, "A Paradigm for Robust Geometric Algorithms," Technical Report TR89-1044, Department of Computer Science, Cornell University, 1989.
9. R. A. JARVIS, On the identification of the convex hull of a finite set of points in the plane, *Inform. Process. Lett.* **2** (1973), 18–21.
10. L. A. LYUSTERNIK (T. J. SMITH, trans.), "Convex Figures and Polyhedra," Dover, New York, 1963.
11. V. J. MILENKOVIC, Verifiable implementations of geometric algorithms using finite precision arithmetic, *in* "Geometric Reasoning" (D. Kapur and J. L. Mundy, Eds.), pp. 377–401, MIT Press, Cambridge, MA, 1989.
12. W. O. J. MOSER, Problems, problems, problems, *Discrete App. Math.* **31** (1991), 201–225.
13. F. P. PREPARATA AND S. J. HONG, Convex hulls of finite sets of points in two and three dimensions, *Comm. ACM* **2** (1977), 87–93.
14. F. P. PREPARATA AND M. I. SHAMOS, "Computational Geometry—An Introduction," Springer-Verlag, New York, 1985.
15. R. SEIDEL, Constructing higher-dimensional convex hulls at logarithmic cost per face, *in* "Proceedings, 18th Annual ACM Symposium on Theory of Computing, 1986," pp. 404–413.
16. E. STEINITZ, Polyeder und Raumeinteilungen, *in* "Encyklopädie der mathematischen Wissenschaften," Band III, Teil 1, 2. Hälfte, IIIAB12, pp. 1–139, 1916.
17. K. SUGIHARA, Topologically consistent algorithms related to convex polyhedra, *in* "Third International Symposium on Algorithms and Computation, Nagoya, 1992," Lecture Notes in Computer Science, Vol. 650, pp. 209–218. Springer-Verlag, Berlin, 1992.
18. K. SUGIHARA AND M. IRI, Two design principles of geometric algorithms in finite-precision arithmetic, *Appl. Math. Lett.* **2** (1989), 203–206.
19. K. SUGIHARA AND M. IRI, Construction of the Voronoi diagram for "one million" generators in single-precision arithmetic, *Proc. IEEE* **80** (1992), 1471–1484.
20. G. SWART, Finding the convex hull facet by facet, *J. Algorithms* **6** (1985), 17–48.