A FAST INCREMENTAL CONVEX HULL ALGORITHM FOR HIGHER DIMENSIONS

by

W.M.Stewart
J.D. Horton

TR90-061 December 1990

Faculty of Computer Science
University of New Brunswick
P.O. Box 4400
Fredericton, N.B. E3B 5A3

Phone: (506) 453-4566
Fax: (506) 453-3566

A Fast Incremental Convex Hull Algorithm for Higher Dimensions

W. M. Stewart

J. D. Horton

Faculty of Computer Science

University of New Brunswick

Fredericton, New Brunswick

Abstract

An online algorithm is described which finds the facets of the convex hull of a set of points in d-dimensional space. For point sets chosen at random from some distribution, the algorithm is $\Theta(nl + F)$ for fixed d, where n is the number of points, l is the expected number of sides of a polygon defined by the intersection of a 2-flat with the polytope, and F is the expected number of facets.

1 Introduction

The convex hull problem is a central one in computational geometry, where the generation of the convex hull is often a preprocessing step for some other process.

Definition 1 *The* convex hull $\mathrm{conv}(S)$ *of a finite set* $S = \{p_1, \ldots, p_n\}$ *in* $\Re^d$ *is the set of all convex combinations of the points of* S*, i.e.*

$$\mathrm{conv}(S) =$$

$$\{\alpha_1 p_1 + \cdots + \alpha_n p_n, \mid \alpha_1 + \cdots + \alpha_n = 1,\ \alpha_i \geq 0\}$$

The convex hull is a convex polytope, or bounded convex polyhedron, that can be represented by its set of facets or highest dimensional faces.

In $\Re^1$, $\mathrm{conv}(S)$ is the closed segment $[\min(S), \max(S)]$, which we represent by its set of facets $\{\min(S), \max(S)\}$. The problem is $\Omega(n)$ and an optimal solution is straightfoward.

In $\Re^2$, $\mathrm{conv}(S)$ is the smallest convex polygon containing S, which we represent by its set of facets or edges. The problem is transformable from sorting, and therefore $\Omega(n \log n)$. The Graham Scan [5] was the first optimal algorithm for $\Re^2$, and a number of subsequent algorithms have been developed with computational advantages for particular distributions.

In $\Re^3$, conv(S) is the smallest 3-dimensional polytope containing S, which we represent by its set of polygonal facets. The $\Omega(n \log n)$ lower bound is inherited from $\Re^2$ and achieved by Preparata and Hong's Divide-and-Conquer algorithm [6].

In $\Re^d$, $d \geq 4$, the number of facets F of a convex hull can rise rapidly, and can be as much as $O(n^{\lfloor d/2 \rfloor})$, but it has been shown that F is no more than $O(n)$ for a remarkably wide range of 'average' distributions (see [4] for a summary).

Chand and Kapur's $O(n \times F)$ Giftwrapping algorithm [3] was the first algorithm developed for higher dimensions. Kallay's Beneath-Beyond algorithm [7] is $O(n \times n^{\lfloor d/2 \rfloor})$ in the worst case, on-line, and maintains the complete facial graph. Seidel's algorithm [8] operates in the dual space, updating the dual facial graph with the addition of each new point, with complexity $\Theta(n^{\lfloor d/2 \rfloor})$ for even d and $O(n \times n^{\lfloor d/2 \rfloor})$ for odd d. Seidel also recognized the significance of the wide variability of the number of facets F for different distributions, and later developed the $O(n^2 + F \log n)$ Shelling algorithm [9].

Stewart in [10] proposed an algorithm which appears from experimental evidence to be $\Theta(n)$ for several symmetric distributions of points. (The overall complexity has not yet been proved.) In fact the algorithm is $\Omega(n \log n)$

since the first step is a sort of the points; the time for this sort is very small compared to the rest of the algorithm for the cases run. The algorithm in this paper is a variant that avoids the sort, and which may be more amenable to an average case analysis. It also seems to run faster for higher dimensions.

2 Preliminaries

Consider a set S of n points in general position in $\Re^d$ for any $d \geq 2$ and $n \geq d+1$.

Definition 2 *An* extreme point $p \in S$ *is not the convex combination of any other two points of* S. *The set of all extreme points is denoted* $\text{ext}(S)$.

Theorem 1 (Brondsted, theorem 7.2) $\mathcal{P} = \text{conv}(S)$ *is the polytope of the extreme points of* S, *i.e.,* $\mathcal{P} = \text{conv}(\text{ext}(S))$.

Definition 3 *A* face $\mathcal{F}$ *of a polytope* $\mathcal{P}$ *is a convex subset such that, for any two distinct points* $y, z \in \mathcal{P}$ *with* $]y,z[\cap \mathcal{F} \neq \emptyset$, *then* $[y,z] \subseteq \mathcal{F}$.

We call a face $\mathcal{F}$ a k-face when $\dim(\mathcal{F}) = k$. The *proper* faces of $\mathcal{P}$ are of dimension 0 through $d-1$, or points through facets, and all lie on the boundary of $\mathcal{P}$.

Theorem 2 (Brondsted, theorem 7.3) *Every proper k-face $\mathcal{F}$ of $\mathcal{P}$ is a k-polytope, and* $\text{ext}(\mathcal{F}) \subset \text{ext}(S)$.

Because S is in general position, every proper k-face is not only a k-polytope, but a k-simplex. We therefore represent a k-face $\mathcal{F}$ by its $k+1$ extreme points $\text{ext}(\mathcal{F}) \subset S$. We are primarily concerned with the $(d-1)$-faces or *facets*, the $(d-2)$-faces or *subfacets*, and the $(d-3)$-faces or *subsubfacets.*

A useful feature of the facial structure of convex polytopes is described by the following well-known theorem.

Theorem 3 *Each subfacet of a polytope is contained in exactly two* neighboring *facets.*

Each simplicial facet contains exactly d simplicial subfacets, and so has exactly d neighbors.

We represent a convex hull by its set of facets $\mathcal{P}$. Each facet $\mathcal{F} \in \mathcal{P}$ is defined by its $d+1$ extreme points, and we also store:

1. $\mathsf{Subf}_1^{\mathcal{F}}, \ldots, \mathsf{Subf}_d^{\mathcal{F}}$: the d subfacets of $\mathcal{F}$.

2. $\mathsf{Neig}_1^{\mathcal{F}}, \ldots, \mathsf{Neig}_d^{\mathcal{F}}$: pointers to neighbors, ordered such that $\mathcal{F} \cap \mathsf{Neig}_i^{\mathcal{F}} = \mathsf{Subf}_i^{\mathcal{F}}$.

3. $\mathsf{Half}^{\mathcal{F}}$: the equation of the halfspace bounded by the hyperplane containing $\mathcal{F}$, and oriented so that $S \subset \mathsf{Half}^{\mathcal{F}}$.

3 The Algorithm

Starting with an initial simplex $\mathcal{P}$, we add the points of S to $\mathcal{P}$ one at a time in random order, maintaining $\mathcal{P}$ as the convex hull of the points so far considered.

First, we describe a procedure by which $\mathcal{P}$ may be updated with a new point p known to be external to $\mathcal{P}$. Second, we describe a method by which we can determine if a new point p is indeed internal or external to $\mathcal{P}$.

3.1 Updating the Convex Hull

$\mathcal{P}$ represents a convex hull by the data structure given above, and p is a point external to $\mathcal{P}$.

Imagine that $\mathcal{P}$ is opaque and that our eye is positioned at p. The update $\mathcal{P} \Leftarrow \mathcal{P} \cup p$ requires five steps:

1. Identify the set of *visible facets* V, where

$$V = \{\mathcal{F} \mid \mathcal{F} \in \mathcal{P},\ p \notin \mathsf{Half}^{\mathcal{F}}\}$$

2. The boundary of V is a set of subfacets called the *horizon* H, where each subfacet of H is shared by one facet of V and one facet of $\mathcal{P} \setminus V$. Create a *cap* C of new facets:

$$C = \{\mathcal{F} \mid \mathcal{F} = f \cup \{p\},\ f \in H\}$$

3. Links must be set between C and the facets of $\mathcal{P} \setminus V$ on the horizon.

4. Links must be set between the facets of C themselves.

5. V must be deleted from $\mathcal{P}$.

We start the update under the assumption that a first visible facet $\mathcal{F}$ is known, and give a method to find this facet in the following section. The first three steps of the update can then be performed as one process:

1. Identify the visible set V by a depth-first search of the neighboring facets of $\mathcal{F}$.

2. During this search, when the neighboring facet $\mathcal{G}$ of a visible facet $\mathcal{F}$ is found to be nonvisible, then subfacet $f = \mathcal{F} \cap \mathcal{G}$ is in the horizon H, so immediately create a new cap facet $\mathcal{T} = f \cup \{p\}$.

3. At the same time, interlink neighboring facets $\mathcal{T}$ and $\mathcal{G}$ across f.

At the conclusion of this depth-first search, we have identified V, created C, and linked C to $\mathcal{P}$.

The interlinking of the cap facets with each other (step 4) is the major step, since there are $(d-1)/2$ as many links to be set between facets of C as there are between C and $\mathcal{P}$.

Lemma 1 *Every facet $\mathcal{T} \in C$ has $d-1$ of its neighbors in C.*

Proof: Only one subfacet of $\mathcal{T}$ does not contain p, i.e., subfacet $f = \mathcal{T} \setminus \{p\}$, and f is shared with a facet of $\mathcal{P}$. The remaining $d-1$ subfacets of $\mathcal{T}$ do contain p, and only facets of C contain p, so $d-1$ neighbors of $\mathcal{T}$ must be in C. □

We set these links between cap facets by a walk around a circuit through neighboring facets, every facet in the circuit containing a particular subsubfacet.

Theorem 4 *There is a circuit through neighboring facets of $\mathcal{P}$, all facets containing the same subsubfacet.*

Proof: Take any facet $\mathcal{F}$ and subsubfacet $\phi \subset \mathcal{F}$. If we consider $\mathcal{F}$ as a $(d-1)$-polytope, then ϕ is a subfacet of $\mathcal{F}$ and by theorem 3 must be

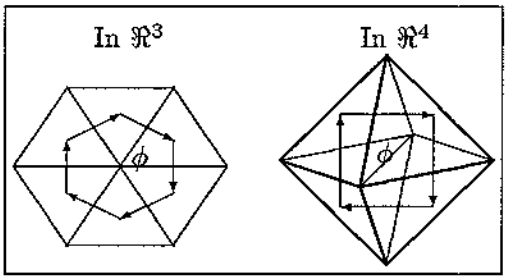

Figure 1: A circuit around subsubfacet ϕ

contained in exactly two distinct facets f and g of $\mathcal{F}$. But, in $\mathcal{P}$, ϕ is contained in exactly two subfacets f and g of $\mathcal{F}$. Therefore exactly two neighbors of $\mathcal{F}$ contain ϕ. □

Figure 1 shows two examples of such a path, one around the subsubfacet or point ϕ of a 3-polytope, and one around a subsubfacet or edge ϕ of a 4-polytope.

Suppose $\mathcal{T}$ is a cap facet, and link $\mathsf{Neig}_j^{\mathcal{T}}$ is not yet set. Then there is a cap facet $\mathcal{U}$ that shares $\mathsf{Subf}_j^{\mathcal{T}}$ with $\mathcal{T}$, and $\mathcal{U}$ must be found in order to set $\mathsf{Neig}_j^{\mathcal{T}}$.

We find $\mathcal{U}$ by a walk around a circuit containing the subsubfacet $\phi = \mathsf{Subf}_j^{\mathcal{T}} \setminus \{p\}$, as follows (refer to figure 2 for an example in $\Re^3$, where $\mathsf{Subf}_j^{\mathcal{T}} = \phi \cup \{p\}$):

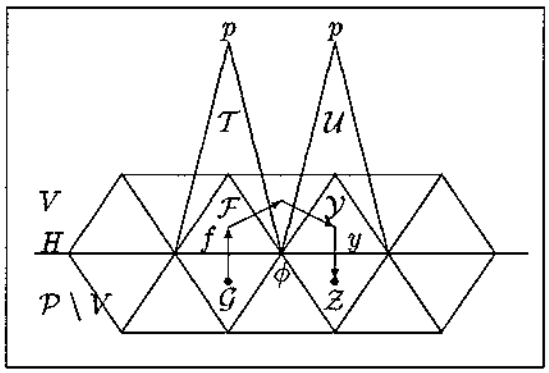

Figure 2: Interlinking Cap Facets in $\Re^3$

1. $\mathcal{T} = f \cup \{p\}$, where f is in the horizon H. Let the visible and non-visible facets sharing f in $\mathcal{P}$ be $\mathcal{F}$ and $\mathcal{G}$ respectively. We know that $p \notin \phi$, therefore $\phi \subset f$, $\phi \subset \mathcal{F}$, and $\phi \subset \mathcal{G}$. Therefore, $\mathcal{F}$ and $\mathcal{G}$ are on a circuit around ϕ, and since this circuit crosses the horizon at f it must cross the horizon again.

2. Walk around ϕ through V, from $\mathcal{F}$ away from $\mathcal{G}$. Stop when the first non-visible facet $\mathcal{Z}$ is encountered. Let the previous (visible) facet on the walk be $\mathcal{Y}$. Let y be the subfacet shared by $\mathcal{Y}$ and $\mathcal{Z}$. Then y is in H, and there is a cap facet $\mathcal{U} = y \cup \{p\}$ linked to $\mathcal{Z}$ across y.

3. Now, $\phi \subset \mathcal{Y}$ and $\phi \subset \mathcal{Z}$, so $\phi \subset y$, and therefore $\phi \subset \mathcal{U}$. As well, $p \in \mathcal{U}$, so the subfacet $\phi \cup \{p\}$ is in $\mathcal{U}$. But $\phi = \mathsf{Subf}_j^{\mathcal{T}} \setminus \{p\}$, or in other words, $\mathsf{Subf}_j^{\mathcal{T}} = \phi \cup \{p\}$. Therefore $\mathsf{Subf}_j^{\mathcal{T}} \subset \mathcal{U}$.

We repeat this procedure to set all empty links between the cap facets, and then delete V from $\mathcal{P}$ (step 5) thereby completing the update $\mathcal{P} \Leftarrow \mathcal{P} \cup p$.

4 The 2-Flat Walk

We now deal with two details postponed earlier. First, since the points of S are added to $\mathcal{P}$ in random order, it is possible that p is interior to $\mathcal{P}$ and no update is actually required. Second, if p is exterior to $\mathcal{P}$, then we require a first visible facet $\mathcal{F}$ to begin the update.

The straightforward approach is costly: scan the facets of $\mathcal{P}$ until one is found which is visible, or until all facets have been found to be non-visible.

The following 2-flat method provides a more efficient solution. Again we make use of our data structure to walk a circuit through neighboring facets:

1. Choose any facet $\mathcal{F} \in \mathcal{P}$. Let q_1 be the centroid of $\mathcal{F}$ and q_2 be the centroid of the whole set S. The three points p, q_1, and q_2 define a 2-flat Q intersecting $\mathcal{F}$.

2. The intersection of a 2-flat and a $(d-1)$-flat is a 1-flat or line. Therefore the intersection of Q with $\mathcal{F}$ is a line segment, and the endpoints of this line segment lie in two subfacets of $\mathcal{F}$. Therefore exactly two neighbors of $\mathcal{F}$ are also intersected by Q, and Q defines a circuit through

neighboring facets of $\mathcal{P}$.

3. Follow the circuit until:

   (a) a facet is encountered that is visible from p; or

   (b) we return to our starting facet $\mathcal{F}$.

In case 3a, p is exterior to $\mathcal{P}$, and we begin the update. In case 3b, p is interior to $\mathcal{P}$.

The first author is presently engaged in an analysis of the complexity of this walk, with a more detailed analysis forthcoming in his doctoral thesis.

At present, experimental results support the conjecture that the average number of steps l taken by the 2-flat walk becomes fewer relative to the number of facets F as the dimension increases. For example, tables 1 and 2 give the total number of steps T and average number of steps l taken by the 2-flat walk during the construction of the convex hulls of sets of points distributed uniformly on the surface of a sphere.

d	F	T	l
3	196	1145	12
4	585	971	10
5	1888	1239	13
6	6884	1290	14
7	23568	1367	15
8	87433	1470	16

Table 1: $|S| = 100$

d	F	T	l
3	396	3391	17
4	1238	3082	16
5	4546	3280	17
6	18407	3631	19
7	72660	4075	21

Table 2: $|S| = 200$

The average length of path l seems to grow slowly with respect to n and F. See table 3 where data for five dimensions is given with the point sets distributed uniformly on the surface of a sphere. Incidently, the calculation

of the data for this table took about six minutes on an IBM-3090 for two sets of data that were then averaged.

n	F	T	l
100	1918	1166	12
200	4533	3312	17
300	7359	6154	21
400	10067	9113	23
500	12928	12131	25
600	15776	16195	27
700	18673	20004	29

Table 3: S in $\Re^5$

5 Complexity

Excluding the 2-flat walk, we present the following analysis of complexity when S is distributed uniformly on the surface of a d-sphere. For this distribution, $S = \text{ext}(S)$.

The number of facet/facet-vertex pairs equals nA, where A is the average number of facets containing a vertex, or $dF = nA$. The complexity of the update step $\mathcal{P} \Leftarrow \mathcal{P} \cup p$ was shown to be $d^3 \times |C|$ in [10]. The overall

complexity of the algorithm is therefore

$$\Theta(nd^3|C|)$$

But for this distribution of S, $|C| = \Theta(A)$, so the overall complexity of the algorithm is

$$\Theta(nd^3A) = \Theta(d^4F)$$

Taking account of l, the average length of the 2-flat walk, we rewrite the complexity as

$$\Theta(nl + d^4F)$$

If F is $\Theta(n)$ and $|\text{ext}(S)| = \Theta(n)$ most of the known algorithms are $\Omega(n^2)$ with the exception of Seidel's dual space algorithm [8] and Stewart's algorithm [10]. This is an important case, because after preprocessing to remove interior points and leaving n exterior vertices, the expected number of faces is often $\Theta(n)$. Dwyer [4] has pointed out that the expected number of vertices and facets are proportional to each other in all studied random distributions. In this case, our algorithm is nearly linear in n, except for the factor l, which is surprisingly small according to our data. In fact, l is almost constant for fixed n as d increases, while F increases exponentially. This indicates that the 2-flat algorithm is expected to be good especially when n is bounded by some as yet to be determined function of d.

References

[1] Brondsted, A. (1983) *An Introduction to Convex Polytopes*, Springer-Verlag, New York.

[2] Buchta, C.; Muller, F.; Tichy, R. F. (1985) *Stochastical Approximation of Convex Bodies*, Math. Annal., 271,225-235.

[3] Chand, D. R.; Kapur, S. S. (1970) *An Algorithm for Convex Polytopes*, J.ACM, 17, 78-86.

[4] Dwyer, R. A. (1990) *Kinder, Gentler Average-Case Analysis for Convex Hulls and Maximal Vectors*, SIGACT News, 21, 64-71.

[5] Graham, R. L. (1972) *An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set*, Inf. Proc. Lett., 1, 132-133.

[6] Preparata, F. P.; Hong, S. J. (1977) *Convex Hulls of Finite Sets in Two and Three Dimensions*, Comm. ACM, 20, 87-93.

[7] Shamos, M. I.; Preparata, F. P. (1985) *Computational Geometry*, Springer-Verlag, New York, 131-134.

[8] Seidel, R. (1981) *A Convex Hull Algorithm Optimal for Point Sets in Even Dimensions*, Tech. Rep. 81-14, Dept. of C.S., Univ. of B.C.,

British Columbia, Canada.

[9] Seidel, R. (1986) *Constructing Higher-Dimensional Convex Hulls at Logarithmic Cost per Face*, Proc. 18th ACM Symp. on Theory of Computing, 404-413.

[10] Stewart, W. M. (1990) *An Algorithm to Find the Facets of the Convex Hull in Higher Dimensions*, Proc. 2nd Canadian Conf. on Computational Geometry, 252-256.