Randomized Quick Hull*

R. Wenger†

October 18, 1995

Abstract

This paper contains a simple, randomized algorithm for constructing the convex hull of a set of n points in the plane with expected running time $O(n \log h)$ where h is the number of points on the convex hull.

Introduction

Determining the convex hull of a set of points is one of the most basic problems in computational geometry. Ron Graham presented the first $O(n \log n)$ algorithm for finding the convex hull of points in the plane in 1972 [12]. The following year, R. Jarvis gave an algorithm whose running time depends on the output size [14]. Jarvis's algorithm runs in $O(nh)$ time where h is the number of points in the convex hull. The next ten years saw many other algorithms for finding convex hulls in the plane most of which run in $O(n \log n)$ time [1, 4, 11, 13, 16]. Some very simple algorithms were proposed which have $O(n)$ expected running time for many distributions of points in the plane (such as points with normal density) [10, 3]. During this period, Avis [2] and Yao [20] proved lower bounds of $\Omega(n \log n)$ on the time to find a convex hull, assuming suitable models of computation.

In 1983, Kirkpatrick and Seidel constructed an "output sensitive" algorithm which runs in $O(n \log h)$ time where h is the number of points on the convex hull [15]. Their algorithm is based on linear time algorithms by Megiddo and Dyer for solving linear programming problems in the plane.

Starting with seminal work by Clarkson, randomized algorithms have played an increasingly important role in computational geometry [8, 7, 9]. These algorithms always produce the correct solution but their running time depends upon random choices made during the execution of the algorithm. As such, they are called Las Vegas random algorithms. Clarkson and Shor in [9] gave randomized incremental algorithms for constructing convex hulls in various dimensions which run in $O(n \log n)$ expected time in $\mathbf{R}^2$ and $\mathbf{R}^3$. They also gave

*To appear in the journal *Algorithmica*.

†Ohio State University, Columbus, OH 43210, U.S.A. (wenger@cis.ohio-state.edu). Supported in part by NSA grant MDA904-93-H-3026 and by the NSF Regional Geometry Institute (Smith College, July 1993) grant DMS-90 13220.

an $O(n \log h)$ expected time algorithm in $\mathbf{R}^3$ which uses Kirkpatrick and Seidel's $O(n \log h)$ algorithm for findng convex hulls in $\mathbf{R}^2$. Seidel in [18] described a simpler randomized incremental algorithm for points in the plane which also runs in $O(n \log n)$ expected time.

This paper contains yet another randomized algorithm for finding the convex hull of points in the plane. However, the expected running time of this algorithm is $O(n \log h)$, matching the worst case running time of the Kirkpatrick and Seidel algorithm. Moreover this algorithm is extremely simple, both to describe and to implement, involving no data structures other than arrays and no presorting of points. The algorithm is a randomized variation of the QUICKHULL algorithm described in [17]. Shafer and Steiger proposed a similar algorithm in a recent paper [19], but without establishing an $O(n \log h)$ bound on the running time of the algorithm independent of the input point set. Differences between our algorithm and the one given by Shafer and Steiger allow us to prove such a bound.

Recently Chan, Snoeyink and Yap published a deterministic variation of QUICKHULL which runs in $O(n \log h)$ time [6]. Their algorithm uses a median find operation in place of randomization but otherwise is quite similar to the algorithm presented in this paper. They generalized their algorithm to higher dimensions. Chan in [5] gave an output-sensitive convex hull algorithm based on ray shooting. His algorithm runs in optimal $O(n \log h)$ time in two and three dimensions.

Randomized Quickhull

The algorithm presented in this paper is called RQ-HULL, an abbreviation for randomized quick hull. Start by assuming that we have two adjacent points p_1 and p_2 on the convex hull and all other points are to the right of the directed line from p_1 to p_2. We can easily find two such points by choosing p_1 with minimum y-coordinate (minimum x-coordinate in case of a tie) and choosing p_2 to be the first point in clockwise angular order around p_1 (farthest from p_1 in case of a tie.) Let P be the set of points to the right of p_1 and p_2.

RQ-HULL takes set P and points p_1 and p_2 as input and reports the extreme points (convex hull vertices) of $P \cup \{p_1, p_2\}$ other than p_1 and p_2. It reports these extreme points in counter-clockwise order around the convex hull of $P \cup \{p_1, p_2\}$.

As in QUICKHULL, RQ-HULL finds an extreme point $q^* \in P$ and divides P into two point sets, P_1 and P_2, using triangle $\triangle(p_1, q^*, p_2)$. Points to the left of the directed line from p_2 to q^* go in one set while points to the right of the directed line from p_1 to q^* go in another. Points contained in triangle $\triangle(p_1, q^*, p_2)$ are not extreme points and can be eliminated. (See Figure 1.) The algorithm is recursively applied to point sets P_1 and P_2 to find their extreme points.

RQ-HULL has two major differences from QUICKHULL. QUICKHULL finds the extreme point q^* which maximizes the area of triangle $\triangle(p_1, q^*, p_2)$. Instead of this deterministic procedure, RQ-HULL uses a randomized algorithm to pick q^*. More specifically, RQ-HULL randomly chooses two points q_1 and

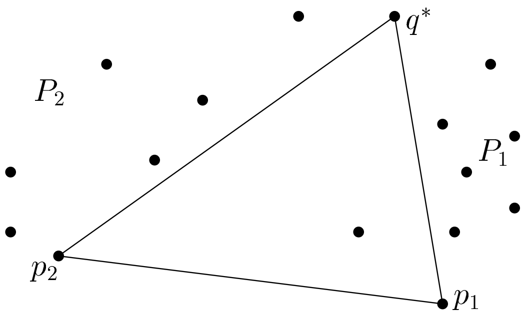

Figure 1: Dividing P by triangle $\triangle(p_1, q^*, p_2)$.

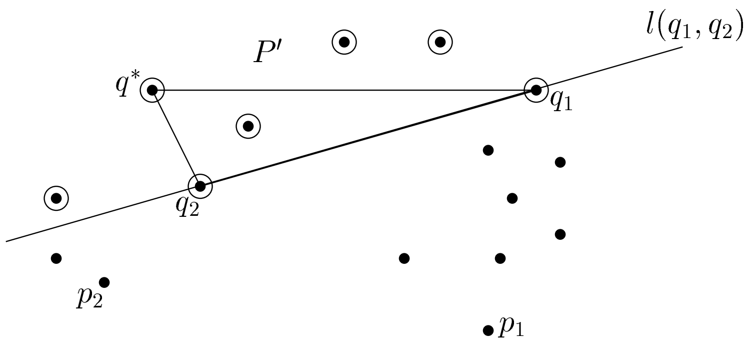

Figure 2: Choosing q^* using randomization.

RQ-HULL(P, p_1, p_2)
/* P = a non-empty set of points to the right of (p_1, p_2) */
/* (i.e., orient(p_1, p_2, p) = Right for all $p \in P$) */
/* Report extreme points of $P \cup \{p_1, p_2\}$ other than p_1 and p_2 */
/* Report points in counter-clockwise order */
1. IF $|P| = 1$ THEN Report $q \in P$; Return;
2. Randomly choose distinct points $q_1, q_2 \in P$, $q_1 \neq q_2$;
3. IF $q_1 \in$ triangle $\triangle(p_1, q_2, p_2)$ THEN RQ-HULL($P - \{q_1\}$, p_1, p_2);
4. ELSE IF $q_2 \in$ triangle $\triangle(p_1, q_1, p_2)$ THEN RQ-HULL($P - \{q_2\}$, p_1, p_2);
5. ELSE
6. $P' \leftarrow \{p \in P : p \in l(q_1, q_2)$ or is separated from p_1 and p_2 by $l(q_1, q_2)$ $\}$;
$(l(q_1, q_2)$ is the line through q_1 and q_2.)
7. Find $q^* \in P'$ which maximizes the area of triangle $\triangle(q_1, q^*, q_2)$;
(Choose the "leftmost" such q^* in case of a tie.)
8. Randomly match points of P, forming a set R of $\lfloor n/2 \rfloor$ pairs;
9. FOR each $(r_1, r_2) \in R$ do
10. IF $r_1 \in$ conv(r_2, q^*, p_1, p_2) THEN $P \leftarrow P - \{r_1\}$;
11. ELSE IF $r_2 \in$ conv(r_1, q^*, p_1, p_2) THEN $P \leftarrow P - \{r_2\}$;
12. $P_1 \leftarrow P_2 \leftarrow \emptyset$;
13. FOR every point $p \in P$,
14. IF orient(p_2, q^*, p) = Left THEN $P_2 \leftarrow P_2 \cup \{p\}$;
15. ELSE IF orient(p_1, q^*, p) = Right THEN $P_1 \leftarrow P_1 \cup \{p\}$;
16. IF $P_1 \neq \emptyset$ THEN RQ-HULL(P_1, p_1, q^*).
17. Report q^*;
18. IF $P_2 \neq \emptyset$ THEN RQ-HULL(P_2, q^*, p_2);

Figure 3: Algorithm RQ-HULL.

q_2 such that points p_1, p_2, q_1, q_2 are the vertices of a convex quadrilateral. It constructs the set P' of points of P which are separated from p_1 and p_2 by the line $l(q_1, q_2)$ through q_1 and q_2, including points lying on $l(q_1, q_2)$. It then picks $q^* \in P'$ to maximize the area of triangle $\triangle(q_1, q^*, q_2)$. (See Figure 2.)

The other difference between QUICKHULL and RQ-HULL is that RQ-HULL attempts to balance the partition of P into P_1 and P_2 by eliminating interior points. RQ-HUll randomly pairs points and checks if one element of the pair lies in the convex hull formed by the other and points p_1, p_2 and q^*. Algorithm RQ-HULL is presented in Figure 3.

A translate of the line $l(q_1, q_2)$ through q_1 and q_2 supports the convex hull of $P \cup \{p_1, p_2\}$ at point q^* so q^* is a vertex of the convex hull. Any point in triangle $\triangle(p_1, q^*, p_2)$ is not on the convex hull. Any point eliminated from P in steps 10 or 11 is also not on the convex hull. All other points are placed in either P_1 or P_2. The convex hull of $P \cup \{p_1, p_2\}$ consists of vertices of the convex hull of $P_1 \cup \{p_1, q^*\}$ and vertices of the convex hull of $P_2 \cup \{q^*, p_2\}$.

Moreover, the former vertices follow the latter ones in counter-clockwise order around the convex hull. By induction, RQ-HULL correctly reports the vertices of the convex hull of $P_1 \cup \{p_1, q^*\}$ followed by q^* followed by the vertices of the convex hull of $P_2 \cup \{q^*, p_2\}$ and so correctly reports the vertices of the convex hull of $P \cup \{p_1, p_2\}$.

Define $T(P, p_1, p_2)$ to be the expected running time of the algorithm RQ-HULL on input (P, p_1, p_2). Define $T(n, h)$ to be the maximum of $T(P, p_1, p_2)$ over all suitable inputs (P, p_1, p_2) where n is the size of P and h is the number of extreme points of $P \cup \{p_1, p_2\}$ other than p_1 and p_2.

PROPOSITION: $T(n, h) = O(n \log h)$.

PROOF: Let P be a set of n points to the right of points p_1 and p_2 and let h be the number of extreme points of $P \cup \{p_1, p_2\}$ other than p_1 and p_2. RQ-HULL starts by randomly choosing two points q_1 and q_2 from P. We divide the analysis of $T(P, p_1, p_2)$ into three different cases based on the relative positions of points q_1, q_2, p_1 and p_2. Let $T_1(P, p_1, p_2)$ be the conditional expected running time of RQHULL given that q_1 lies in triangle $\triangle(p_1, q_2, p_2)$. Let $T_2(P, p_1, p_2)$ be the conditional expected running time of RQHULL given that q_2 lies in triangle $\triangle(p_1, q_1, p_2)$. Finally, let $T_3(P, p_1, p_2)$ be the conditional expected running time of RQHULL given that $q_1 \notin \triangle(p_1, q_2, p_2)$ and $q_2 \notin \triangle(p_1, q_1, p_2)$.

If q_1 lies in triangle $\triangle(p_1, q_2, p_2)$, the algorithm recursively calls RQ-HULL$(P - \{q_1\}, p_1, p_2)$. Thus $T_1(P, p_1, p_2)$ is bounded by $T(n-1, h) + c$ for some constant c. Using an inductive assumption that $T(n-1, h) \leq c_1(n-1)\log h$ for some constant c_1 gives $T_1(P, p_1, p_2) \leq c_1(n-1)\log h + c_0 \leq c_1 n \log h$ for suitably large c_1. Similarly, $T_2(P, p_1, p_2) \leq c_1 n \log h$.

Now consider the case that $q_1 \notin \triangle(p_1, q_2, p_2)$ and $q_2 \notin \triangle(p_1, q_1, p_2)$. Let $p(n_1, h_1, n_2, h_2)$ be the probability that P_i contains n_i vertices and h_i extreme points, $i = 1, 2$, given that $q_1 \notin \triangle(p_1, q_2, p_2)$ and $q_2 \notin \triangle(p_1, q_1, p_2)$. Steps 6-15 in RQ-HULL take $O(n)$ time. All the other steps take constant time excluding the time spent in the recursive calls. Thus

$$T_3(P, p_1, p_2) \leq \sum p(n_1, h_1, n_2, h_2)(T(n_1, h_1) + T(n_2, h_2)) + c_2 n + c_3$$

for some constants c_2 and c_3. Using an inductive assumption that $T(n', h') \leq c_1 n' \log h'$ for $n' < n$ and $h' \leq h$ gives

$$T_3(P, p_1, p_2) \leq \sum p(n_1, h_1, n_2, h_2)(c_1 n_1 \log h_1 + c_1 n_2 \log h_2) + c_2 n + c_3.$$

Since $n_1 + n_2 \leq n - 1$ and $h_1 + h_2 \leq h - 1$,

$$n_1 \log h_1 + n_2 \log h_2 \leq (n-1)\log(h-1).$$

If we add the conditions that $\max(n_1, n_2) \leq (31/32)n$ and $h_1 \geq h_2$, then

$$n_1 \log h_1 + n_2 \log h_2 \leq (31/32) n \log h_1 + (1/32) n \log h_2.$$

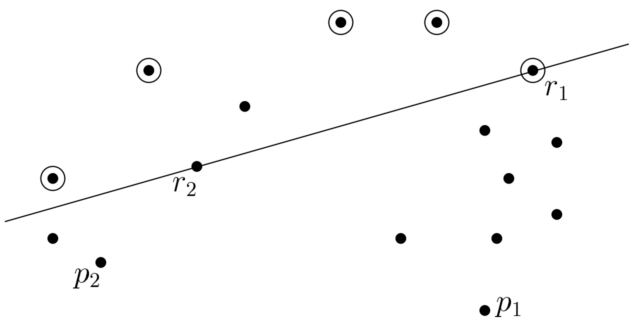

Figure 4: The set of points $v(r_1, r_2)$.

A simple application of derivatives shows that the right hand side of this equation is maximized when $h_1 = (31/32)(h-1)$ and $h_2 = (h-1)/32$ where it equals

$$n \log(h-1) - n \log \left(\left(\frac{32}{31} \right)^{31/32} 32^{1/32} \right).$$

Setting $\alpha = \log((32/31)^{31/32} 32^{1/32})$, we get

$$n_1 \log(h_1 - 1) + n_2 \log(h_2 - 1) \quad \leq \quad n(\log(h-1) - \alpha).$$

The same bounds hold under the conditions that $\max(n_1, n_2) \leq (31/32)n$ and $h_1 \leq h_2$.

In the lemma following this proof, we show that $\max(n_1, n_2) \leq (31/32)n$ with probability at least 1/60. Splitting the summand depending upon whether $\max(n_1, n_2)$ is greater or less than $(31/32)n$ we get

$$\begin{aligned} T_3(P, p_1, p_2) &\leq \frac{59}{60} c_1 (n-1) \log(h-1)) + \frac{1}{60} c_1 n (\log(h-1) - \alpha) + c_2 n + c_3 \\ &\leq c_1 n \log h - \frac{1}{60} c_1 \alpha n + c_2 n + c_3. \end{aligned}$$

This is bounded by $c_1 n \log h$ for suitably large c_1.

Since $T_1(P, p_1, p_2)$, $T_2(P, p_1, p_2)$ and $T_3(P, p_1, p_2)$ are each bounded by $c_1 n \log h$ and $T(P, p_1, p_2)$ is a weighted average of these three conditional expectations, $T(P, p_1, p_2)$ is also bounded by $c_1 n \log h$. This bound applies to any suitable input (P, p_1, p_2) where P contains n points and $P \cup \{p_1, p_2\}$ contains h extreme points other than p_1 and p_2, so $T(n, h) = O(n \log h)$. □

LEMMA: *The probability that* $\max(|P_1|, |P_2|) \leq (31/32)n$ *is at least 1/60.*

PROOF: Let Π' be the set of pairs (r_1, r_2) of points in P where r_1, r_2, p_1, p_2 are the vertices of a convex quadrilateral. Let Π'' be the set of all other pairs

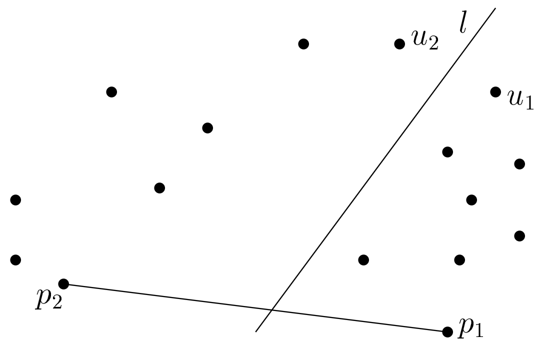

Figure 5: Splitting P with line l.

of points in P. For each $(r_1, r_2) \in \Pi'$, let $v(r_1, r_2)$ be the extreme points of $P \cup \{p_1, p_2\}$ which are separated from p_1 and p_2 by a line through r_1 and r_2. (See Figure 4.) Extreme points which lie on the line through r_1 and r_2 should also be included in $v(r_1, r_2)$. The extreme points $v(r_1, r_2)$ form a contiguous set of vertices along the convex hull of $P \cup \{p_0, p_1\}$. Note that the vertex of the convex hull supporting the line parallel to the line through r_1 and r_2 lies in $v(r_1, r_2)$. Note also that for any vertex $q \in v(r_1, r_2)$, either the triangle q, p_1, p_2 divides r_1 from r_2 or points q, r_1, r_2, p_1, p_2 are not the vertices of a convex pentagon. For each $(r_1, r_2) \in \Pi''$, let $v(r_1, r_2)$ be the entire set of extreme vertices of $P \cup \{p_1, p_2\}$ other than p_1 and p_2.

For each extreme point q of $P \cup \{p_1, p_2\}$ assign a weight $w(q)$ equal to the number of pairs $(r_1, r_2) \in \Pi' \cup \Pi''$ where $q \in v(r_1, r_2)$. Let q^* be the point found in step 6 of the algorithm. We claim that the probability that $w(q^*) \geq n^2/16$ is at least $1/4$.

Draw a line l, possibly passing through a point of P, which crosses line segment (p_1, p_2) and divides P into two equal size sets of exactly $\lfloor n/2 \rfloor$ points. (See Figure 5.) Line l separates or passes through an endpoint of at least $n^2/4$ pairs of points. Let (u_1, u_2) where $u_1, u_2 \in P \cup \{p_1, p_2\}$ be the edge of the convex hull of $P \cup \{p_1, p_2\}$ which is intersected by l. For each pair of points (r_1, r_2) split by l, either u_1 or u_2 or both are in $v(r_1, r_2)$. Thus either $w(u_1)$ or $w(u_2)$ or both are greater than $n^2/8$. Without loss of generality, assume $w(u_1) \geq n^2/8$. Let γ be the largest set of contiguous hull vertices containing u_1 such that $w(q) \geq n^2/16$ for all $q \in \gamma$.

If set γ contains all the extreme points of $P \cup \{p_1, p_2\}$ other than p_1 and p_2, then clearly $q^* \in \gamma$ and so $w(q^*) \geq n^2/16$. Assume set γ is bounded by two extreme points s_1 and s_2 with weights less than $n^2/16$. There are at most $n^2/8$ pairs (r_1, r_2) where $s_1 \in v(r_1, r_2)$ or $s_2 \in v(r_1, r_2)$. Thus, for at least $n^2/8$ of the $n^2/4$ pairs (r_1, r_2) split by l, the set $v(r_1, r_2)$ is a subset of γ. Note that these $n^2/8$ pairs are all in Π', not Π''. The probability that (q_1, q_2) chosen in step 2 are from this set of $n^2/8$ is at least $(n^2/8)/|\Pi'|$ which is bounded from below by $(n^2/8)/\binom{n}{2} \geq 1/4$. Thus the probability that $v(q_1, q_2) \subseteq \gamma$ is at least $1/4$. Since $q^* \in v(q_1, q_2)$, the probability that $q^* \in \gamma$ and thus $w(q^*) \geq n^2/16$ is at least $1/4$. The case where γ is bounded by a single extreme point is handled similarly.

Assume that $w(q^*) \geq n^2/16$. The probability that $q^* \in v(r_1, r_2)$ for any pair of points (r_1, r_2) is at least $(n^2/16)/\binom{n}{2} = (1/8)\frac{n}{n-1}$. Set R constructed in step 7 contains $\lfloor n/2 \rfloor$ pairs of points. The expected number of pairs $(r_1, r_2) \in R$ where $q^* \in v(r_1, r_2)$ is at least $(1/8)\frac{n}{n-1}\lfloor n/2 \rfloor \geq n/16$. Similarly, the expected number of pairs of R where $q^* \notin v(r_1, r_2)$ is at most $\lfloor n/2 \rfloor - n/16$. By Markov's inequality, the probability that more than $(15/14)(\lfloor n/2 \rfloor - n/16)$ pairs $(r_1, r_2) \in R$ have the property that $q^* \notin v(r_1, r_2)$ is at most $14/15$. Thus the probability that at least $\lfloor n/2 \rfloor - (15/14)(\lfloor n/2 \rfloor - n/16) \geq n/32$ pairs $(r_1, r_2) \in R$ have the property that $q^* \in v(r_1, r_2)$ is at least $1/15$.

Let (r_1, r_2) be any pair of points in R. If r_1, r_2, q^*, p_1, p_2 are not the vertices of a convex pentagon, either r_1 or r_2 will be deleted in steps 10-11. If r_1, r_2, q^*, p_1, p_2 are the vertices of a convex pentagon and $q^* \in v(r_1, r_2)$, then q^* must lie between r_1 and r_2 because the line through r_1 and r_2 separates q^* from p_1 and p_2. In either case, if $q^* \in v(r_1, r_2)$, then P_1 will not contain both r_1 and r_2, nor will P_2. Thus if $n/32$ pairs $(r_1, r_2) \in R$ have the property that $q^* \in v(r_1, r_2)$, then the sizes of both P_1 and P_2 and the value of $\max(|P_1|, |P_2|)$ must be less than or equal to $(31/32)n$.

The probability that $w(q^*) \geq n^2/16$ is at least $1/4$. Given that $w(q^*) \geq n^2/16$, the probability that R contains $n/32$ pairs (r_1, r_2) where $q^* \in v(r_1, r_2)$ is at least $1/15$. Thus the probability that $\max(|P_1|, |P_2|) \leq (31/32)n$ is at least $1/60$.

□

Conclusion

RQ-HULL runs in expected $O(n \log h)$ time, matching the $O(n \log h)$ deterministic running time of Kirkpatrick and Seidel's algorithm. The hidden constant in the analysis of the $O(n \log h)$ running time of RQ-HULL is quite large. This constant seems to reflect the weakness of the given probabilistic argument, not the running time of RQ-HULL.

Steps 8-11 in RQ-HULL speed the algorithm by eliminating interior points. What happens if these steps are omitted? The algorithm still reports the convex hull of $P \cup \{p_0, p_1\}$ but we were unable to bound the expected running time. We suspect the algorithm still runs in $O(n \log n)$ but not necessarily $O(n \log h)$ expected time. RQ-HULL without steps 8-11 is similar to an algorithm described by Shafer and Steiger in [19].

Acknowledgements

Thanks to Kennith Supowit for helpful discussions on bounding recurrence relations.

References

[1] Akl, S. G., and Toussaint, G. T. A fast convex hull algorithm. *Inform. Process. Lett. 7*, 5 (1978), 219–222.

[2] Avis, D. Comments on a lower bound for convex hull determination. *Inform. Process. Lett. 11* (1980), 126.

[3] Bentley, J. L., and Shamos, M. I. Divide-and-conquer for linear expected time. *Inform. Process. Lett. 7* (1978), 87–91.

[4] Bykat, A. Convex hull of a finite set of points in two dimensions. *Inform. Process. Lett. 7* (1978), 296–298.

[5] Chan, T. Ouput-sensitive results on convex hulls, extreme points, and related problems. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.* (1995), pp. 10–19.

[6] Chan, T., Snoeyink, J., and Yap, C. K. Ouput-sensitive construction of polytopes in four dimensions and clipped Voronoi diagrams in three. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms* (1995), pp. 282–291.

[7] Clarkson, K. L. New applications of random sampling in computational geometry. *Discrete Comput. Geom. 2* (1987), 195–222.

[8] Clarkson, K. L. A randomized algorithm for closest-point queries. *SIAM J. Comput. 17* (1988), 830–847.

[9] Clarkson, K. L., and Shor, P. W. Applications of random sampling in computational geometry, II. *Discrete Comput. Geom. 4* (1989), 387–421.

[10] Devroye, L. P. How to reduce the average complexity of convex hull finding algorithms. *Comput. Math. Appl. 7* (1981), 299–308.

[11] Eddy, W. F. A new convex hull algorithm for planar sets. *ACM Trans. Math. Softw. 3* (1977), 398–403 and 411–412.

[12] Graham, R. L. An efficient algorithm for determining the convex hull of a finite planar set. *Inform. Process. Lett. 1* (1972), 132–133.

[13] Green, P. J., and Silverman, B. W. Constructing the convex hull of a set of points in the plane. *Comput. J. 22* (1979), 262–266.

[14] Jarvis, R. A. On the identification of the convex hull of a finite set of points in the plane. *Inform. Process. Lett. 2* (1973), 18–21.

[15] Kirkpatrick, D. G., and Seidel, R. The ultimate planar convex hull algorithm? *SIAM J. Comput. 15* (1986), 287–299.

[16] Koplowitz, J., and Jouppi, D. A more efficient convex hull algorithm. *Inform. Process. Lett. 7* (1978), 56–57.

[17] Preparata, F. P., and Shamos, M. I. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.

[18] Seidel, R. Backwards analysis of randomized geometric algorithms. In *New Trends in Discrete and Computational Geometry*, J. Pach, Ed., vol. 10 of *Algorithms and Combinatorics*. Springer-Verlag, 1993, pp. 37–68.

[19] Shafer, L., and Steiger, W. Randomizing optimal geometric algorithms. In *Proc. 5th Canad. Conf. Comput. Geom.* (Waterloo, Canada, 1993), pp. 133–138.

[20] Yao, A. C. A lower bound to finding convex hulls. *J. ACM 28* (1981), 780–787.