Random Sampling, Halfspace Range Reporting, and Construction of ($\leq k$)-Levels in Three Dimensions*

Timothy M. Chan†

September 16, 1999

Abstract

Given n points in three dimensions, we show how to answer halfspace range reporting queries in $O(\log n+k)$ expected time for an output size k. Our data structure can be preprocessed in optimal $O(n \log n)$ expected time. We apply this result to obtain the first optimal randomized algorithm for the construction of the ($\leq k$)-level in an arrangement of n planes in three dimensions. The algorithm runs in $O(n \log n+nk^2)$ expected time. Our techniques are based on random sampling. Applications in two dimensions include an improved data structure for "k nearest neighbors" queries, and an algorithm that constructs the order-k Voronoi diagram in $O(n \log n + nk \log k)$ expected time.

Key words. computational geometry, range searching, levels in arrangements, nearest neighbor searching, Voronoi diagrams, randomized data structures, randomized algorithms

AMS subject classifications. 68U05, 68Q25, 68P05, 52B30

Abbreviated title. Halfspace range reporting and ($\leq k$)-levels

1 Introduction

1.1 Halfspace range reporting

Let P be a set of n points in d-dimensional space $\mathbb{R}^d$. We consider the problem of preprocessing P so that given any query halfspace γ, one can quickly report all points in $P \cap \gamma$. A vast literature in computational geometry has been devoted to this fundamental problem called *halfspace range reporting*, a special case of *range searching* [4, 37, 49, 53, 55]. Here are some of the major known results.

First, halfspace range reporting in the planar case ($d = 2$) was solved optimally by Chazelle, Guibas, and Lee [28]. Their data structure takes linear space and answers a query in $O(\log n+k)$ time, where k is the number of reported points. The preprocessing can be accomplished in $O(n \log n)$ time using Chazelle's algorithm for *convex layers* [22]. Unfortunately, the approach does not generalize to higher dimensions.

*A preliminary version of this paper appeared in *Proc. 39th IEEE Sympos. Found. Comput. Sci.*, 1998. This work was done while the author was at the Department of Mathematics and Computer Science, University of Miami.

†Department of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada, tmchan@math.uwaterloo.ca

In the $d = 3$ case, Chazelle and Preparata [29] gave a method for answering halfspace range reporting queries in optimal time $O(\log n + k)$. The space complexity is $O(n \log^2 n \log\log n)$, as noted by Clarkson and Shor [32]. The preprocessing is $O(n \log^c n)$ for a small constant c and uses shallow *levels* in arrangements (see Section 1.2). Aggarwal, Hansen, and Leighton [10] subsequently improved the space bound to $O(n \log n)$ while maintaining optimal query time. However, the preprocessing is more complex, exploiting advanced techniques such as *planar separators*; a deterministic version requires near-cubic time, while a Monte Carlo version needs $O(n \log^2 n \log\log n)$ time. In particular, it remained open whether $O(n \log n)$ preprocessing time is attainable. If one is willing to sacrifice optimality in the query bound, then a simple method [25] involving a tree of *planar point location* structures solves the problem with $O(k \log^2 n)$ query time and $O(n \log n)$ preprocessing time and space.

For a larger fixed dimension $d \geq 4$, Clarkson and Shor [32] used shallow *cuttings* to achieve $O(\log n + k)$ query time with $O(n^{\lfloor d/2 \rfloor + \varepsilon})$ preprocessing time and space, where ε is an arbitrarily small positive constant. Matoušek [48] obtained a certain *partition theorem* that implies a data structure with $O(n \log n)$ preprocessing time, $O(n \log\log n)$ space, and $O(n^{1-1/\lfloor d/2 \rfloor} \log^{O(1)} n + k)$ query time. (This method can be specialized to handle the $d = 3$ case; the query time appears to be $O((\log n)^{O(\log\log n)} + k)$.) Tradeoffs between preprocessing and query time were also described by Matoušek. These higher-dimensional results were conjectured to be optimal up to n^{ε} or polylogarithmic factors.

The first result of this paper is to close the existing gap for halfspace range reporting for the case $d = 3$. In Section 2, we give a relatively simple, randomized (Las Vegas) method that can answer queries in $O(\log n + k)$ expected time. This data structure requires $O(n \log n)$ space and can be preprocessed in $O(n \log n)$ expected time. The expectation here is with respect to the random choices made by the preprocessing; it is assumed that the given query halfspaces are independent of these choices. Both time bounds on preprocessing and querying are optimal.

Our approach is to consider random samples of various sizes of the dual planes and examine the *canonical triangulations* of their (≤ 0)-*levels*. With good chances, it turns out that the answer to a query with output size k can be found within some *conflict list* of the triangulation for a sample of size approximating n/k; the expected size of this list is $O(k)$. The preprocessing of all such conflict lists is done by imitating a known algorithm for the (≤ 0)-level (i.e., *convex hull* in primal space). We should remark that ideas like random sampling, canonical triangulations, and conflict lists are hardly new in this area; what is not apparent is how they can be put together to derive the optimal result. For instance, Clarkson and Shor's halfspace range reporting structure [32] uses already a top-down form of random sampling, which seems inherently suboptimal; we avoid such difficulties by adopting a bottom-up approach instead.

Our result has several important applications. For example, we can now preprocess n points in the Euclidean plane in $O(n \log n)$ expected time, such that *"k nearest neighbors" queries* can be answered in $O(\log n + k)$ expected time; this and the related *circular range reporting* problem are two of the most basic proximity problems, dating back to the beginning of computational geometry. For another application, we can enumerate the k *bichromatic closest pairs* of n planar points in $O(n \log n + k)$ expected time. See Section 2.3. One less obvious application—the construction of levels in arrangements—is the second main topic of this paper.

1.2 Levels in arrangements

Given a set H of n hyperplanes in d-dimensional space $\mathbb{R}^d$ and $0 \leq k \leq n$, define the region

$$\mathrm{lev}_k(H) \;=\; \{q \in \mathbb{R}^d : \text{at most } k \text{ hyperplanes of } H \text{ pass strictly below } q\}.$$

The $(\leq k)$*-level* in the arrangement of H is the collection of faces in the arrangement that are contained in $\mathrm{lev}_k(H)$. The related notion of the k*-level* can be defined as the collection of faces contained in the boundary of $\mathrm{lev}_k(H)$. Levels in arrangements are among the most well-studied, yet most puzzling, geometric structures in both their combinatorial and computational aspects [8, 37, 43, 53].

Our initial focus is on $(\leq k)$-levels. The main combinatorial question here was answered by Clarkson and Shor [32], who showed via a beautiful random sampling argument that the $(\leq k)$-level in a fixed dimension d can have at most $O(n^{\lfloor d/2\rfloor}k^{\lceil d/2\rceil})$ faces. This bound is tight in the worst case. The computational complexity of $(\leq k)$-levels was also essentially resolved for $d = 2$ and $d \geq 4$. Everett, Robert, and van Kreveld [42] gave an $O(n\log n + nk)$-time algorithm in the plane (see also [12]). Mulmuley [52] gave a randomized algorithm for dimensions four and higher with expected running time $O(n^{\lfloor d/2\rfloor}k^{\lceil d/2\rceil})$. These two results are optimal in view of Clarkson and Shor's bound and the trivial $\Omega(n\log n)$ lower bound.

The important case $d = 3$ however remains an open problem, even though a number of algorithms have been developed. A randomized algorithm of Mulmuley [52] for instance runs in expected time $O(nk^2\log(n/k))$, which is just a logarithmic factor away from optimal. Agarwal, de Berg, Matoušek, and Schwarzkopf [2] proposed a different randomized algorithm with expected running time $O(n\log^3 n + nk^2)$, which is optimal for sufficiently large k. In Section 3, we settle the complexity of the three-dimensional case completely by giving a randomized algorithm with an $O(n\log n + nk^2)$ expected time bound; this is optimal for all values of k. Before, this time bound was known only for the special case where the input planes are *non-redundant*, i.e., they all bound the (≤ 0)-level [9, 52].

Our approach deviates from the previous algorithms of Mulmuley and Agarwal *et al.* in that it does not follow the randomized incremental paradigm [32, 53]. Instead, the idea (at a high level) is to choose a random sample of size n/k and use the canonical triangulation of its arrangement to divide the problem into roughly $O(n/k)$ subproblems of average size $O(k)$. These subproblems are created from conflict lists—computable by halfspace range reporting—and are then solved by brute force in $O(k^3)$ average time. The analysis of our algorithm is based on a *shallow cutting lemma* of Matoušek.

A similar approach can be taken to compute the k-level, although optimality is not yet attained for this problem. First of all, the combinatorial problem of determining the worst-case size of the k-level is wide open (the dual is related to the famous k*-set problem* [8, 11, 37]). Significant development occurred recently in the planar case $d = 2$ when Dey [34] improved the long-standing upper bound of (roughly) $O(n\sqrt{k})$ to $O(nk^{1/3})$, but the current best lower bound remains $\Omega(n\log k)$ [41]. For $d = 3$, the most recent upper bound is $O(nk^{5/3})$ [1, 35]. For higher dimensions, the current upper bound is only slightly better than the $O(n^{\lfloor d/2\rfloor}k^{\lceil d/2\rceil})$ bound for the larger $(\leq k)$-level. There is a case where the exact complexity of the k-level is known: if $d = 3$ and all input planes are non-redundant, then the k-level has size $\Theta(nk)$ for $k \leq n/2$ [32, 46]. The k-level in this case is related to the *order-*k *Voronoi diagram* of n points in the Euclidean plane—a natural extension to one of the most fundamental and useful geometric structures, the *Voronoi diagram.*

Some of the algorithmic results obtained by Agarwal *et al.* [2] on the k-level can be improved by our techniques. Specifically, their expected time bound of $O(n\log^2 n + nk^{1/3}\log^{2/3} n)$ for the construction of k-levels in the plane can be reduced to $O(n\log n + nk^{1/3}\log^{2/3} k)$. Furthermore, their $O(n\log^3 n + nk\log n)$ algorithm for order-k Voronoi diagrams in the plane can be sped up to run in $O(n\log n + nk\log k)$ expected time. These results are described in Section 3.3.

2 Halfspace Range Reporting in $\mathbb{R}^3$

2.1 Preliminaries

Let H be a given set of n (nonvertical) hyperplanes in $\mathbb{R}^d$. For simplicity, we assume that they are in general position; standard perturbation techniques [38, 17] can be applied to remove this assumption. The halfspace range reporting problem by *duality* [37, 53] is equivalent to the following: preprocess H so that given a query point q, one can quickly report all hyperplanes of H below q. We will actually solve a slightly harder problem: preprocess H so that given a query vertical line ℓ and a number k, one can quickly report the k lowest hyperplanes along ℓ (i.e., hyperplanes defining the k lowest intersections with ℓ). The connection between halfspace range reporting queries and such *"k lowest hyperplanes" queries* will be explained later.

Given a subset $R \subset H$, the (≤ 0)-level in the arrangement of R (also called the *lower envelope* of R) is a convex polyhedron. Computing this polyhedron is equivalent to constructing the *convex hull* [37, 53, 55] by duality. For $d = 3$, several $O(|R|\log|R|)$-time algorithms are known [32, 54], among the simplest of which are based on the randomized incremental paradigm.

Let $\mathrm{CT}_0(R)$ denote the collection of (closed) full-dimensional simplices in the *canonical triangulation* of the (≤ 0)-level, as defined for instance by Clarkson [31] (also called the *bottom-vertex triangulation*). The precise definition of this triangulation is not important to us except in the proof of the sampling lemma below. The only facts we need is that the triangulation is linear-time constructible and that the simplices in $\mathrm{CT}_0(R)$ are all vertical cylinders containing the point $(0, \ldots, 0, -\infty)$.

Thus, a vertical line ℓ hits precisely one simplex Δ in $\mathrm{CT}_0(R)$, if one ignores degenerate cases. For $d = 3$, this simplex Δ can be identified in $O(\log|R|)$ time after an $O(|R|\log|R|)$-time preprocessing, as the problem projects down to *planar point location* [37, 55].

Given a simplex Δ, the *conflict list* H_Δ is defined as the set of all hyperplanes of H intersecting Δ. The following two sampling results are needed in the analysis of our data structure. Both follow from the general probabilistic technique by Clarkson and Shor [32]. (The first is often used in analyses of randomized convex hull algorithms.)

Lemma 2.1 *Let $1 \leq r \leq n$ and consider a random sample $R \subset H$ of size r.*

(i) *The expected value of the sum $\sum_\Delta |H_\Delta|$ over all simplices $\Delta \in \mathrm{CT}_0(R)$ is $O(r^{\lfloor d/2\rfloor} \cdot n/r)$.*

(ii) *For any fixed vertical line ℓ, the expected value of $|H_\Delta|$ for the simplex $\Delta \in \mathrm{CT}_0(R)$ hit by ℓ is $O(n/r)$.*

2.2 The data structure for $d = 3$

We take a common approach called *bottom-up sampling* by Mulmuley [53]. Choose a random permutation $h_1, \ldots, h_n$ of the set H. Define $R_i = \{h_1, \ldots, h_{2^i}\}$ for $i = 0, 1, \ldots, \log n$ (without loss of generality, say n is a power of 2; logarithms are in base 2). The result is a sequence ("hierarchy") of random samples

$$R_0 \subset R_1 \subset \cdots \subset R_{\log n} = H,$$

where $|R_i| = 2^i$. Our basic data structure is simple: it consists of location structures for $\mathrm{CT}_0(R_i)$, along with the conflict list H_Δ for all the simplices $\Delta \in \mathrm{CT}_0(R_i)$. For each R_i, the space needed is $O(\sum_{\Delta \in \mathrm{CT}_0(R_i)} |H_\Delta|)$, which has expected value $O(n)$ by Lemma 2.1(i). The total expected space is therefore $O(n \log n)$. We can guarantee that space is within a constant factor of this bound by repeating for an expected constant number of trials.

We now describe how to build this data structure efficiently. First the location structures can all be constructed in time $O(\sum_{i=1}^{\log n} |R_i| \log |R_i|) = O(n \log n)$. The nontrivial part is the computation of the conflict lists. It turns out that this can be done by just modifying a known randomized algorithm for convex hulls in $\mathbb{R}^3$. We consider here Clarkson and Shor's original method [32], which maintains global conflict information incrementally. (Note that online methods based on *history* [53] are not suitable for our purposes.)

Specifically, at the 2^i-th step of Clarkson and Shor's method (in the version described by Mulmuley's or Motwani and Raghavan's text [53, 50]), we not only have all vertices of the (≤ 0)-level in the arrangement of R_i, but in addition, have for each plane $h \in H$ a pointer to some vertex lying above h (if one exists). By a graph search, we can generate all vertices lying above each h. As a result, we have for each vertex v a list of all planes of H below v. Given a simplex $\Delta \in \mathrm{CT}_0(R_i)$ with vertices v_1, v_2, v_3, the conflict list H_Δ is just the union of the list of planes of H below the v_j's.

Clarkson and Shor's method runs in $O(n \log n)$ expected time. The extra work done at the 2^i-th step to produce the conflict lists is $O(\sum_{\Delta \in \mathrm{CT}_0(R_i)} |H_\Delta|)$, which has expected value $O(n)$ by Lemma 2.1(i). Hence, the whole preprocessing of our data structure can be performed in $O(n \log n)$ expected time.

We now describe the basic query algorithm. Given vertical line ℓ and number k as input parameters, the algorithm is usually able to find the k lowest planes of H along ℓ but occasionally may report failure instead. The probability of failure is controlled by a third input parameter $\delta > 0$.

Algorithm ANSWER-QUERY(ℓ, k, δ)

1. let $i = \lceil \log \lceil n\delta/k \rceil \rceil$
2. identify the simplex $\Delta \in \mathrm{CT}_0(R_i)$ cut by ℓ
3. if $|H_\Delta| > k/\delta^2$ then return "failed"
4. if fewer than k planes of H_Δ intersect $\ell \cap \Delta$ then return "failed"
5. return the k lowest planes of H_Δ along ℓ

The algorithm is correct: what is returned in line 5 is precisely the k lowest planes of H along ℓ if failure is not reported in line 4. The running time of ANSWER-QUERY() is $O(\log n + k/\delta^2)$, since line 2 takes $O(\log n)$ time by planar point location, and lines 4 and 5 take $O(k/\delta^2)$ time if failure is not reported in line 3. (Line 5 is an instance of what Chazelle referred to as *filtering search* [23].)

We now bound the failure probability for any fixed choice of ℓ, k, and δ. By Lemma 2.1(ii), the expected value of $|H_\Delta|$ is $O(n/|R_i|) = O(k/\delta)$, so by Markov's inequality, the probability that H_Δ

exceeds k/δ^2 is $O(\delta)$. Thus, line 3 reports failure with probability $O(\delta)$. On the other hand, letting q denote the k-th lowest intersection of H along ℓ, we see that line 4 reports failure only if $q \notin \Delta$, or equivalently, $q \notin \mathrm{lev}_0(R_i)$. This is true only if one of the k planes below q is chosen to be in the sample R_i. As q is independent of R_i, this can happen with probability at most $k|R_i|/n = O(\delta)$.

We can summarize our result as follows:

Theorem 2.2 *In $O(n \log n)$ expected time, one can preprocess n planes in $\mathbb{R}^3$ into a randomized data structure of $O(n \log n)$ size, such that there is a procedure with the following behavior. Given any fixed vertical line ℓ, number k, and $\delta > 0$, the procedure either reports the k lowest planes along ℓ or reports failure. The probability of failure is $O(\delta)$, but the procedure always runs within $O(\log n + k/\delta^2)$ time.*

2.3 Consequences

It is desirable to modify the data structure to have a query algorithm that never fails. This modification can be done in two stages. First we observe that having three independent versions of the basic data structure can reduce the failure probability to $O(\delta^3)$ in Theorem 2.2:

Corollary 2.3 *In $O(n \log n)$ expected time, one can preprocess n planes in $\mathbb{R}^3$ into a randomized data structure of $O(n \log n)$ size, such that there is a procedure satisfying the criteria stated in Theorem 2.2 but with failure probability $O(\delta^3)$.*

Next we apply the basic query algorithm on a sequence of choices for the parameter δ in order to guarantee success.

Corollary 2.4 *In $O(n \log n)$ expected time, one can preprocess n planes in $\mathbb{R}^3$ into a randomized data structure of $O(n \log n)$ size, such that any "k lowest planes" query can be answered in $O(\log n + k)$ expected time.*

Proof: Let $\delta_i = 2^{-i}$. Run the procedure of Corollary 2.3 for $\delta = \delta_1, \delta_2, \ldots$ until it succeeds. Let X_i denote the 0-1 random variable with value 1 when the procedure fails for $\delta = \delta_i$. The total running time is bounded asymptotically by

$$(\log n + k/\delta_1^2) \; + \; \sum_{i>1} (\log n + k/\delta_i^2)\, X_{i-1},$$

which has expected value $O(\sum_{i>1} (\log n + k/\delta_i^2)\, \delta_{i-1}^3) \; = \; O(\log n + k)$. □

We still have to explain how halfspace range reporting queries reduces to "k lowest planes" queries. This can be done by a standard technique of "guessing" the parameter k.

Corollary 2.5 *In $O(n \log n)$ expected time, one can preprocess n points in $\mathbb{R}^3$ into a randomized data structure of $O(n \log n)$ size, such that a halfspace range reporting query can be answered in $O(\log n + k)$ expected time, where k is the number of points reported.*

Proof: Recall that a halfspace range reporting query in dual space corresponds to finding all k planes below a given point q; the value of k is not known in advance. Let $k_i = 2^i \log n$ and ℓ be the vertical line at q. This task can be accomplished by searching for the k_i-th lowest plane along ℓ for $i = 1, 2, \ldots$ until such a plane lies above q; then we simply examine the k_i lowest planes along ℓ and report those that are actually below q. The expected running time is asymptotically bounded by

$$(\log n + k_1) + \sum_{k_{i-1} < k} (\log n + k_i) \;=\; O(\log n + k),$$

because of Corollary 2.4. $\square$

Remarks:

1. While the preprocessing and query time are optimal, they are only expected bounds; furthermore, the space complexity can be improved. Using advanced tools and a larger preprocessing time, Appendix A.3 gives a modification of our data structure that is deterministic and uses only $O(n \log\log n)$ space.

2. It is worthwhile to compare Chazelle, Guibas, and Lee's optimal method [28] with the specialization of our method in two dimensions. Ours seems easier to implement as convex layers are not involved.

3. Higher-dimensional extensions are possible, although we do not see any significant improvements.

By a standard lifting map, Corollaries 2.4 and 2.5 imply a new method for *"k nearest neighbors"* and *circular range reporting queries* [10, 16, 25, 37, 53, 55, 57] in the Euclidean plane:

Corollary 2.6 *In $O(n \log n)$ expected time, one can preprocess n point sites in $\mathbb{R}^2$ into a randomized data structure of $O(n \log n)$ size, such that the k closest (or farthest) sites of a given point can be found in $O(\log n + k)$ expected time.*

Corollary 2.7 *In $O(n \log n)$ expected time, one can preprocess n point sites in $\mathbb{R}^2$ into a randomized data structure of $O(n \log n)$ size, such that all sites inside (or outside) a given circle can be reported in $O(\log n + k)$ expected time, where k is the output size.*

Another proximity application in the Euclidean plane is an optimal randomized algorithm for enumerating the k *bichromatic closest pairs* [45]. The author [19] gave a randomized reduction of this problem to a reporting problem, which in turn reduces to answering n offline halfspace range reporting queries in $\mathbb{R}^3$, where the total output size is k.

Corollary 2.8 *Given n-point sets P and Q in $\mathbb{R}^2$, one can enumerate the k closest (or farthest) pairs from $P \times Q$ (not necessarily in sorted order) in expected time $O(n \log n + k)$.*

3 $(\leq k)$-Levels in $\mathbb{R}^3$

3.1 Preliminaries

Let H be a given set of n hyperplanes in $\mathbb{R}^d$ in general position. Our goal in this section is to construct the facial structure of the $(\leq k)$-level in the arrangement of H. As before, the key concept is the canonical triangulation. Given a subset $R \subset H$, define $\mathrm{CT}(R)$ to be the collection of full-dimensional simplices in the canonical triangulation of the faces in the arrangement of R. Define $\mathrm{CT}_k(R)$ similarly for the faces of the $(\leq k)$-level in the arrangement of R.

It is somewhat more involved to compute conflict lists for simplices arising from the construction of $(\leq k)$-levels, e.g., the simplices are not necessarily vertical cylinders now. For this reason, we define $\Delta^* = \mathrm{conv}(\Delta \cup \{(0,\ldots,0,-\infty)\})$, the *vertical extension* of a simplex Δ. For $d = 3$, the extended conflict list H_{Δ^*} can be computed in $O(\log n + |H_{\Delta^*}|)$ expected time after $O(n \log n)$-time preprocessing by Corollary 2.5: listing all planes of H below a given point reduces to halfspace range reporting, and if $v_1, \ldots, v_4$ denote the vertices of Δ, then H_{Δ^*} is just the union of the lists of planes of H below the v_j's.

One other important definition is the following: a simplex Δ is *relevant* if it intersects the region $\mathrm{lev}_k(H)$. The sampling lemma below is stated implicitly in the proof of Matoušek's *shallow cutting lemma* [48] (see Appendix A.2) and is needed in the analysis of our algorithm. Matoušek's proof uses probabilistic arguments of Chazelle and Friedman [27]; see Appendix A.1 for more details.

Lemma 3.1 *Let $1 \leq r \leq n$ and $q = kr/n + 1$. Let $f(n)$ be a* regular *function, i.e., a nondecreasing function satisfying $f(2n) = O(f(n))$. Consider a random sample $R \subset H$ of size r. The expected value of the sum $\sum_\Delta f(|H_\Delta|)$ over all relevant simplices $\Delta \in \mathrm{CT}(R)$ is bounded by $O(r^{\lfloor d/2 \rfloor} q^{\lceil d/2 \rceil} f(n/r))$.*

3.2 The algorithm for $d = 3$

The basic outline of our $(\leq k)$-level algorithm is quite simple: we pick a random sample $R \subset H$ of size n/k, generate all relevant simplices of $\mathrm{CT}(R)$, compute the $(\leq k)$-level within each such simplex Δ, and then combine the solutions.

The $(\leq k)$-level in the arrangement of H within a relevant simplex Δ can be constructed by the following procedure:

Algorithm SOLVE-SUBPROBLEM(Δ), where Δ is a relevant simplex

1. compute conflict list H_{Δ^*}
2. construct the $(\leq k)$-level in the arrangement of H_{Δ^*} by brute force
3. clip the resulting faces to Δ

The correctness of the procedure is easy to see (as $\mathrm{lev}_k(H) \cap \Delta = \mathrm{lev}_k(H_{\Delta^*}) \cap \Delta$). Line 1 takes $O(\log n + |H_{\Delta^*}|)$ expected time, whereas lines 2 and 3 take $O(|H_{\Delta^*}|^3)$ time by a brute force method: construct the entire arrangement of H_{Δ^*} [39] and extract the desired substructure. Since Δ is relevant, we can bound $|H_{\Delta^*}|$ by $k + |H_\Delta|$. Therefore, SOLVE-SUBPROBLEM(Δ) runs in expected time $O(\log n + (k + |H_\Delta|)^3)$.

We now describe how to generate all relevant simplices of $\mathrm{CT}(R)$. Observe that any relevant simplex of $\mathrm{CT}(R)$ must belong to $\mathrm{CT}_k(R)$ (since $\mathrm{lev}_k(H) \subseteq \mathrm{lev}_k(R)$). Assume that $\mathrm{CT}_k(R)$ is available. To generate all relevant simplices of $\mathrm{CT}_k(R)$, we perform a graph search. Say that two simplices of $\mathrm{CT}_k(R)$ are *adjacent* if they share a common facet. Notice that a simplex can be adjacent to at most four other simplices in $\mathrm{CT}_k(R)$. Consider the following procedure:

Algorithm GENERATE-SUBPROBLEMS(Δ), where $\Delta \in \mathrm{CT}_k(R)$

```
1.  mark Δ
2.  SOLVE-SUBPROBLEM(Δ)
3.  for each unmarked simplex Δ' adjacent to Δ do
4.     let ξ be the facet shared by Δ and Δ'
5.     if ξ intersects lev_k(H) then GENERATE-SUBPROBLEMS(Δ')
```

The test in line 5 can be easily accomplished from the information obtained from line 2. An initial relevant simplex $\Delta_0 \in \mathrm{CT}_k(R)$ is easy to find (e.g., pick any simplex from $\mathrm{CT}_0(R)$). If all simplices are initially unmarked and GENERATE-SUBPROBLEMS(Δ_0) is called, then all relevant simplices of $\mathrm{CT}_k(R)$ (and thus of $\mathrm{CT}(R)$) will be marked because the region $\mathrm{lev}_k(H)$ is connected.

Combining the solutions is straightforward. We just take all the output from line 2 of GENERATE-SUBPROBLEMS() in this process and "stitch" these substructures together to get the complete ($\leq k$)-level in the arrangement of H. (The implementation details of stitching are not entirely trivial though; for example, one needs to match features along common facets between simplices. This can be done in linear time, for instance, by labeling planes as integers and faces as tuples and then applying a radix sort.)

Excluding the $O(n \log n)$ preprocessing time, the expected time bound for the calls to GENERATE-SUBPROBLEMS() is

$$O\left(\sum_{\Delta}(\log n + (k + |H_\Delta|)^3)\right),$$

where the sum is taken over all relevant simplices Δ of $\mathrm{CT}(R)$. Lemma 3.1 tells us that the expectation of the above expression with respect to R for a sample size $r = n/k$ is

$$O((n/k)(\log n + k^3)) \;=\; O(n \log n + nk^2).$$

For the final piece of the algorithm, we still have to explain how $\mathrm{CT}_k(R)$ is obtained in the first place. This can be done by recursively constructing the ($\leq k$)-level in the arrangement of R. The total expected running time of our algorithm is then given by the recurrence

$$T_k(n) \;=\; T_k(n/k) \;+\; O(n \log n + nk^2),$$

which solves to $T_k(n) = O(n \log n + nk^2)$, assuming (without loss of generality) that $k \geq 2$.

Theorem 3.2 *The ($\leq k$)-level in an arrangement of n planes in $\mathbb{R}^3$ can be constructed in $O(n \log n + nk^2)$ expected time.*

Remarks:

1. Although our algorithm is optimal for worst-case output, the running time is not the best possible if the ($\leq k$)-level has size smaller than $\Theta(nk^2)$. It remains open to devise an optimal *output-sensitive* algorithm [6, 18, 51].

2. Derandomization seems to require modification to our algorithm because the sample size is quite large. Appendix A.2 sketches an optimal deterministic method for ($\leq k$)-levels for large k (specifically, $\log k = \Omega(\log n)$). This method works in any fixed dimension but uses advanced tools.

3. Specialization to $d = 2$ yields a new algorithm for ($\leq k$)-levels in the plane with expected running time $O(n \log n + nk)$.

3.3 Application to k-levels

The algorithm can be modified to construct the k-level in the arrangement of H for $d = 3$. To get good results, we need to have available an algorithm for constructing the k-level in subcubic time, say $O(f(n))$ time (randomized or deterministic).

We will make SOLVE-SUBPROBLEM(Δ) output the faces of the k-level clipped to Δ. This can be done by changing line 2 to construct just the k-level in the arrangement of H_{Δ^*} in $O(f(|H_{\Delta^*}|))$ time. In GENERATE-SUBPROBLEMS(Δ), the test in line 5 can also be performed in $O(f(|H_{\Delta^*}|))$ time, since it is equivalent to deciding whether ξ intersects the region beneath the k-level in the arrangement of H_{Δ^*}.

Thus, ignoring the $O(n \log n)$ preprocessing, the expected time to construct the entire k-level is

$$O\left(\sum_{\Delta}(\log n + f(k + |H_\Delta|))\right),$$

where the sum is taken over all relevant simplices Δ of $\mathrm{CT}(R)$. By Lemma 3.1, the expectation with respect to R for a sample size $r = n/k$ and a regular function f is $O((n/k)(\log n + f(k)))$.

Note that the recursive call to compute $\mathrm{CT}_k(R)$ is not needed now: by Theorem 3.2, the ($\leq k$)-level in the arrangement of R can be constructed in expected time $O((n/k)\log(n/k) + (n/k)k^2)$. Since $f(k) = \Omega(k^2)$, the overall complexity is $O(n \log n + (n/k)f(k))$.

The same strategy applies to k-levels in dimension $d = 2$. We therefore have shown:

Theorem 3.3 *Let $\mathcal{H}$ be a class of lines in $\mathbb{R}^2$ or planes in $\mathbb{R}^3$. Suppose that the k-level in the arrangement of any set $H \subset \mathcal{H}$ of size n can be computed in $O(f(n))$ expected time, where $f(n)$ is a regular function. Then the k-level can be constructed in $O(n \log n + (n/k)f(k))$ expected time.*

Specifically, Agarwal, de Berg, Matoušek, and Schwarzkopf [2] have obtained the following bounds via randomized incremental construction: for lines in $\mathbb{R}^2$, $f(n) = n^{4/3} \log^{2/3} n$; and for planes in $\mathbb{R}^3$ that are tangent to the unit paraboloid, $f(n) = n^2 \log n$. The first case takes into account Dey's recent combinatorial bound on the k-level (or more precisely, on the $O(\log n)$ consecutive levels below the k-level). The second case occurs in the construction of *order-k Voronoi diagrams* [13, 37, 53, 55, 57] in the Euclidean plane by the standard lifting map; see Table 1 for some previous algorithms. For arbitrary planes in $\mathbb{R}^3$, we have $f(n) = n^{8/3+\varepsilon}$ by an algorithm of Agarwal and Matoušek [6] (with the known combinatorial bound [35]). Hence, Theorem 3.3 implies:

Corollary 3.4 *The k-level in an arrangement of n lines in $\mathbb{R}^2$ can be constructed in expected time $O(n \log n + nk^{1/3} \log^{2/3} k)$.*

year	time bound	references
1982	$O(nk^2 \log n)^*$	Lee [46]
1986	$O(n^3)^*$	Edelsbrunner, O'Rourke, and Seidel [39]
1986	$O(nk\sqrt{n} \log n)$	Edelsbrunner [36]
1987	$O(n^2 + nk \log^2 n)$	Chazelle and Edelsbrunner [26]
1987	$O(n^{1+\varepsilon} k)$ rand.	Clarkson [30]
1989	$O(n \log n + nk^2)^*$	Aggarwal, Guibas, Saxe, and Shor [9]
1991	$O(n \log n + nk^2)^*$ rand.	Mulmuley [52]
1992	$O(nk \log^2 n + nk^2)$ rand. online	Aurenhammer and Schwarzkopf [14]
1993	$O(n \log n + nk^3)^*$ rand. online	Boissonnat, Devillers, and Teillaud [15]
1995	$O(n^{1+\varepsilon} k)$	Agarwal and Matoušek [6]
1998	$O(n \log^3 n + nk \log n)$ rand.	Agarwal, de Berg, Matoušek, and Schwarzkopf [2]
now	$O(n \log n + nk \log k)$ rand.	this paper
	$O(nk \log^2 k\, (\log n / \log k)^{O(1)})$	appendix

Table 1: History of algorithms for the order-k Voronoi diagram in $\mathbb{R}^2$ ($k \leq n/2$). Year refers to date of journal publication. (Bounds marked * actually apply to all diagrams of order 1 to k.)

Corollary 3.5 *The order-k Voronoi diagram of n point sites in $\mathbb{R}^2$—i.e., the planar subdivision where two points belong to the same region iff they have the same set of k closest (or farthest) sites—can be constructed in expected time $O(n \log n + nk \log k)$.*

Corollary 3.6 *The k-level in an arrangement of n planes in $\mathbb{R}^3$ can be constructed in expected time $O(n \log n + nk^{5/3+\varepsilon})$.*

Remarks:

1. Theorem 3.3 can be viewed as an algorithmic version of a combinatorial result of Agarwal, Aronov, Chan, and Sharir [1], who showed basically that if the k-level in $\mathbb{R}^2$ or $\mathbb{R}^3$ has worst-case complexity $O(f(n))$, then the k-level has compexity $O((n/k)f(k))$.

2. Corollary 3.4 compares favorably with an output-sensitive algorithm of Edelsbrunner and Welzl [40], which runs in time $O(n \log n + f \log^2 n)$ for an f-face output. Only slight improvements of this algorithm were known: Cole, Sharir, and Yap [33] discussed how to reduce the second term to $O((n + f) \log^2 k)$ and the author [18] showed how to reduce the first term to $O(n \log f)$.

3. Corollary 3.5 is currently the best result for the construction of a single order-k Voronoi diagram in the plane. It is optimal for $k = O(\log n / \log \log n)$. If $f(n)$ can be improved to $O(n^2)$, then Theorem 3.3 would imply a randomized algorithm running in optimal $O(n \log n + nk)$ expected time for any $k \leq n/2$.

4. It is interesting to note that while in the past, levels in arrangements have been used to solve the halfspace range reporting problem, we have taken just the reverse approach, using halfspace range reporting to construct levels. Is there a more direct way to construct levels with the same efficiency?

Update. After the conference version of this paper, Ramos [56] has recently improved Agarwal *et al.*'s algorithm [2] for the planar order-k Voronoi diagram and has obtained $f(n) = n^2 2^{O(\log^* n)}$, thereby reducing the expected time bound in Corollary 3.5 to $O(n \log n + nk2^{O(\log^* k)})$. For k-levels of lines in the plane, the author [21] has noted an improved bound $f(n) = n^{4/3}$, which reduces the expected time bound in Corollary 3.4 to $O(n \log n + nk^{1/3})$. There are also some new breakthroughs in output-sensitive algorithms for two-dimensional levels by Har-Peled [44] and the author [20, 21].

A Appendix

A.1 Proof sketch of Lemma 3.1

As Lemma 3.1 is the key in proving the optimality of our randomized ($\leq k$)-level algorithm, a proof sketch of the lemma may be appropriate to give the reader an idea why the probabilistic bound holds. As mentioned earlier, the details of the proof are essentially embedded in the paper by Matoušek [48]. We highlight the main points here in somewhat different notation.

First the following definition is helpful: we say that a simplex Δ is j-*good* if it is contained inside the region $\mathrm{lev}_j(H)$. An immediate observation is that any relevant simplex Δ is $(k + |H_\Delta|)$-good. It turns out that bounding the number of j-good simplices is easier than bounding the number of relevant simplices:

> The expected number of j-good simplices $\Delta \in \mathrm{CT}(R)$ is $O((r/n)^d n^{\lfloor d/2 \rfloor} j^{\lceil d/2 \rceil})$.

The reason is this: as can be seen from the definition of canonical triangulations, the number of j-good simplices of $\mathrm{CT}(R)$ is linear in the number of vertices of $\mathrm{CT}(R)$ that are contained in $\mathrm{lev}_j(H)$. We know that $\mathrm{lev}_j(H)$ has $O(n^{\lfloor d/2 \rfloor} j^{\lceil d/2 \rceil})$ vertices, and the probability that a fixed vertex in the arrangement of H appears in the arrangement of R is $O((r/n)^d)$.

The second step is to observe that the conflict size $|H_\Delta|$ is usually of the order of n/r. For instance, arguments by Clarkson and Shor [32] show that the average conflict size is expected to be $O(n/r)$. We will actually need a stronger result by Chazelle and Friedman [27] (see also [7]), stating that the number of conflict sizes of the order of tn/r decreases exponentially with t. Specifically, one can prove the following statement from our earlier bound on j-good simplices:

> The expected number of j-good simplices $\Delta \in \mathrm{CT}(R)$ with $|H_\Delta| \geq (t-1)n/r$ is $O(2^{-t}(r/n)^d n^{\lfloor d/2 \rfloor} j^{\lceil d/2 \rceil})$.

As a consequence, we can then bound a similar quantity for relevant simplices if we just substitute $k + tn/r$ for j:

> The expected number of relevant simplices $\Delta \in \mathrm{CT}(R)$ with $(t-1)n/r \leq |H_\Delta| \leq tn/r$ is $O(2^{-t}(r/n)^d n^{\lfloor d/2 \rfloor} (k + tn/r)^{\lceil d/2 \rceil})$.

Finally, we can bound the expectation of the sum $\sum_\Delta f(|H_\Delta|)$ over all relevant simplices $\Delta \in \mathrm{CT}(R)$ asymptotically by

$$\sum_{t=1}^{\infty} 2^{-t}(r/n)^d n^{\lfloor d/2 \rfloor}(k + tn/r)^{\lceil d/2 \rceil} f(tn/r) \;=\; O((r/n)^d n^{\lfloor d/2 \rfloor}(k + n/r)^{\lceil d/2 \rceil} f(n/r)),$$

since f is regular. Notice that the above expression is identical to $O(r^{\lfloor d/2 \rfloor} q^{\lceil d/2 \rceil} f(n/r))$.

A.2 A deterministic algorithm for $(\leq k)$-levels in $\mathbb{R}^d$

In the second appendix, we briefly consider derandomization of our $(\leq k)$-level algorithm in an arbitrary fixed dimension. An optimal bound is obtained only when k is sufficiently large. The approach employs a deterministic version of Lemma 3.1, derived by Matoušek [48] using tools such as the *method of conditional probabilities* and *ε-approximations.*

Lemma A.1 (Matoušek's Shallow Cutting Lemma) *Let $1 \leq r \leq n$ and $q = kr/n+1$. One can cover $\mathrm{lev}_k(H)$ by a collection of $O(r^{\lfloor d/2 \rfloor} q^{\lceil d/2 \rceil})$ simplices such that $|H_\Delta| \leq n/r$ for each simplex Δ. Furthermore, the simplices have disjoint interiors. They can be constructed in $O(n \log r)$ time, provided that $r \leq n^\alpha$ for a sufficiently small constant $\alpha > 0$ depending on d.*

Worst-case efficiency demands us to use small values of r, so we will construct the $(\leq k)$-level by divide-and-conquer: (i) pick $r = \min\{n/k,\, n^\alpha\}$ and find the collection of simplices by the above lemma; (ii) for each simplex Δ, compute conflict list H_{Δ^*}; (iii) construct the $(\leq k)$-level in the arrangement of H_{Δ^*} by recursion; and (iv) finally combine the solutions by stitching.

Step (ii) requires some explanation. From the discussion of our three-dimensional algorithm, we see that a conflict list can be computed by answering a constant number of halfspace range reporting queries. With known results [48], this requires $O(n^\beta + |H_{\Delta^*}|)$ time after $O(n \log n)$-time preprocessing, for a constant $\beta \approx 1 - 1/\lfloor d/2 \rfloor$. Notice that if $|H_{\Delta^*}|$ exceeds $k + |H_\Delta|$ the simplex Δ is not relevant and need not be considered in step (iii). So, we can ensure that a query runs within time $O(n^\beta + k + n/r)$.

Accounting all the costs, we derive this recurrence for the total running time:

$$T_k(n) \;=\; O(n \log n) \;+\; O(r^{\lfloor d/2 \rfloor} q^{\lceil d/2 \rceil})\, (n^\beta + T_k(k + n/r)).$$

Assuming that $\alpha < (1-\beta)/\lfloor d/2 \rfloor$ without loss of generality, we can simplify the above to

$$T_k(n) \;=\; O(n \log n) \;+\; O(r^{\lfloor d/2 \rfloor})\, T_k(k + n/r).$$

Our base case is when $n^{1-\alpha} \leq k$. Here, $r = n/k$ and we solve the subproblems directly by constructing entire arrangements in $T_k(2k) = O(k^d)$ time [39]; the overall time bound is

$$T_k(n) \;=\; O(n \log n + (n/k)^{\lfloor d/2 \rfloor} k^d) \;=\; O(n \log n + n^{\lfloor d/2 \rfloor} k^{\lceil d/2 \rceil}).$$

If $n^{1-\alpha} > k$, then $r = n^\alpha$ and the recurrence becomes

$$T_k(n) \;=\; O(n \log n) \;+\; O(n^{\alpha \lfloor d/2 \rfloor})\, T_k(2n^{1-\alpha}),$$

which solves to

$$T_k(n) \;=\; O\left((n \log n + n^{\lfloor d/2 \rfloor} k^{\lceil d/2 \rceil}) \left(\frac{\log n}{\log k} \right)^{O(1)} \right) \;=\; O\left(n^{\lfloor d/2 \rfloor} k^{\lceil d/2 \rceil} \left(\frac{\log n}{\log k} \right)^{O(1)} \right).$$

Theorem A.2 *The $(\leq k)$-level in an arrangement of n hyperplanes in $\mathbb{R}^d$ can be constructed deterministically in time $O(n^{\lfloor d/2 \rfloor} k^{\lceil d/2 \rceil} (\log n / \log k)^{O(1)})$.*

A similar approach works for k-levels and order-k Voronoi diagrams. We mention that for the latter problem in two dimensions, the best deterministic result can be achieved with Chazelle and Edelsbrunner's $O(n^2 \log^2 n)$ bound [26] for the base cases:

Theorem A.3 *The order-k Voronoi diagram of n point sites in $\mathbb{R}^2$ can be constructed deterministically in time $O(nk \log^2 k\, (\log n/\log k)^{O(1)})$.*

Remarks:

1. The use of the shallow cutting lemma to construct levels deterministically has been noted before in a paper by Agarwal, Efrat, and Sharir [3]; however, our deterministic bounds appear new.
2. Theorem A.2 is worst-case optimal if $k = \Omega(n^\varepsilon)$ for some constant $\varepsilon > 0$. For small k, optimal derandomization for arbitrary dimensions appears difficult, as can be seen from Chazelle's work on convex hulls [24].

Update. The author [20, 21] has recently improved the bound in Theorem A.3 slightly to $O(nk \log^{1+\varepsilon} n\, (\log n/\log k)^{O(1)})$.

A.3 A deterministic data structure for halfspace range reporting in $\mathbb{R}^3$

In this final appendix, we revisit the halfspace range reporting problem in $\mathbb{R}^3$ and give a deterministic data structure with space $O(n \log\log n)$ and worst-case query time $O(\log n + k)$. The space bound improves the one by Aggarwal, Hansen, and Leighton [10].

The approach is based on our randomized method, but to obtain a successful derandomization, we need to replace the (≤ 0)-levels of the samples (and their canonical triangulations) with suitable structures. Shallow cuttings serve exactly this purpose, but first a slight variant is stated for convenience:

Lemma A.4 *Let $d = 3$. One can cover $\mathrm{lev}_k(H)$ by a collection $\mathcal{T}_k$ of $O(n/k)$ simplices such that $|H_\Delta| = O(k)$ for each $\Delta \in \mathcal{T}_k$. Furthermore, the simplices have disjoint interiors, each containing $(0, 0, -\infty)$.*

Proof: Let Ξ be a collection of simplices satisfying Lemma A.1 with $r = n/k$. Without loss of generality, we may assume that each simplex $\Delta \in \Xi$ is relevant; thus, each vertex has at most $|H_\Delta| + k \leq 2k$ planes of H below it. Now, let U be the union of Ξ and define $\mathcal{T}_k$ to be a triangulation of U into vertical cylinders. Because a plane in H_Δ must lie below one of the vertices of Δ, we have $|H_\Delta| \leq 6k$ for each $\Delta \in \mathcal{T}_k$. □

The chief ingredient to reduce space from $O(n \log n)$ to $O(n \log\log n)$ is bootstrapping with an $O(n)$-space structure that has query time $O(n^\beta + k)$ for a constant $\beta < 1$. Known results on the simplex range searching [47] imply that $\beta \approx 2/3$ is possible in $\mathbb{R}^3$; the time bound applies to "k lowest planes" queries as well [5].

Define a sequence $k_1, k_2, \ldots$ by the formula $k_i = \left\lfloor k_{i-1}^{1/\beta} \right\rfloor$, starting with a constant and ending when the term reaches n. Evidently, there are $O(\log\log n)$ terms. Our data structure consists of a hierarchy of triangulations constructed by Lemma A.4: $\mathcal{T}_{k_1}, \mathcal{T}_{k_2}, \ldots$ In addition, we build a location structure for $\mathcal{T}_{k_i}$ and store the linear-space range searching structure for each conflict list H_Δ ($\Delta \in \mathcal{T}_{k_i}$). The storage requirement is $O((n/k)k) = O(n)$ for each $\mathcal{T}_{k_i}$, and $O(n \log\log n)$ overall.

To find the k lowest planes of H along a vertical line ℓ, let index i satisfy $k_{i-1} \leq k < k_i$ and determine the simplex $\Delta \in \mathcal{T}_{k_i}$ hit by ℓ; by projection, this is a planar point location problem and takes $O(\log n)$ time. As the answer is a subset of the conflict list H_Δ (since $\mathcal{T}_{k_i}$ covers $\mathrm{lev}_k(H)$), we can use the range searching structure for H_Δ to answer the query. The total query time is

$$O(\log n + |H_\Delta|^\beta + k) \;=\; O(\log n + k_i^\beta + k) \;=\; O(\log n + k).$$

Theorem A.5 *Given n planes in $\mathbb{R}^3$, there exists a data structure of $O(n \log\log n)$ size, such that any "k lowest planes" query can be answered in $O(\log n + k)$ time deterministically.*

Consequences on halfspace range reporting, k nearest neighbors, and circular range reporting can be immediately derived, as in Corollaries 2.5, 2.6, and 2.7.

Remarks:

1. The preprocessing time is prohibitively large (though polynomial), so our randomized method is still more practical.

2. It remains an open problem to bring down the space complexity to linear while maintaining optimal query time. (Note that if the same k is used in all queries and is given in advance, then our approach achieves linear space.)

Update. Theorem A.5 has also been independently shown by Ramos [56] using ideas from the conference version of this paper, with a better expected preprocessing time of $O(n \log n)$. His proof is more complicated though and appears to be only of theoretical interest.

References

[1] P. K. Agarwal, B. Aronov, T. M. Chan, and M. Sharir. On levels in arrangements of lines, segments, planes, and triangles. *Discrete Comput. Geom.*, 19:315–331, 1998.

[2] P. K. Agarwal, M. de Berg, J. Matoušek, and O. Schwarzkopf. Constructing levels in arrangements and higher order Voronoi diagrams. *SIAM J. Comput.*, 27:654–667, 1998.

[3] P. K. Agarwal, A. Efrat, and M. Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. In *Proc. 11th ACM Sympos. Comput. Geom.*, pages 39–50, 1995.

[4] P. K. Agarwal and J. Erickson. Geometric range searching and its relatives. To appear in *Discrete and Computational Geometry: Ten Years Later* (B. Chazelle, J. E. Goodman, and R. Pollack, ed.), AMS Press.

[5] P. K. Agarwal and J. Matoušek. Ray shooting and parametric search. *SIAM J. Comput.*, 22:764–806, 1993.

[6] P. K. Agarwal and J. Matoušek. Dynamic half-space range reporting and its applications. *Algorithmica*, 13:325–345, 1995.

[7] P. K. Agarwal, J. Matoušek and O. Schwarzkopf. Computing many faces in arrangements of lines and segments. *SIAM J. Comput.*, 27:491–505, 1998.

[8] P. K. Agarwal and M. Sharir. Arrangements and their applications. To appear in *Handbook of Computational Geometry* (J. Urrutia and J. Sack, ed.), North-Holland.

[9] A. Aggarwal, L. J. Guibas, J. Saxe, and P. W. Shor. A linear-time algorithm for computing the Voronoi diagram of a convex polygon. *Discrete Comput. Geom.*, 4:591–604, 1989.

[10] A. Aggarwal, M. Hansen, and T. Leighton. Solving query-retrieval problems by compacting Voronoi diagrams. In *Proc. 22nd ACM Sympos. Theory Comput.*, pages 331–340, 1990.

[11] A. Andrzejak and E. Welzl. k-sets and j-facets: a tour of discrete geometry. Manuscript, 1997.

[12] T. Asano and T. Tokuyama. Topological walk revisited. In *Proc. 6th Canad. Conf. Comput. Geom.*, pages 1–6, 1994.

[13] F. Aurenhammer. Voronoi diagrams: a survey of a fundamental geometric data structure. *ACM Comput. Surveys*, 23:345–405, 1991.

[14] F. Aurenhammer and O. Schwarzkopf. A simple on-line randomized incremental algorithm for computing higher order Voronoi diagrams. *Int. J. Comput. Geom. Appl.*, 2:363–381, 1992.

[15] J.-D. Boissonnat, O. Devillers, and M. Teillaud. A semidynamic construction of higher-order Voronoi diagrams and its randomized analysis. *Algorithmica*, 9:329–356, 1993.

[16] J. L. Bentley and H. A. Maurer. A note on Euclidean near neighbor searching in the plane. *Inform. Process. Lett.*, 8:133–136, 1979.

[17] I. Emiris and J. Canny. A general approach to removing degeneracies. *SIAM J. Comput.*, 24:650–664, 1995.

[18] T. M. Chan. Output-sensitive results on convex hulls, extreme points, and related problems. *Discrete Comput. Geom.*, 16:369–387, 1996.

[19] T. M. Chan. On enumerating and selecting distances. In *Proc. 14th ACM Sympos. Comput. Geom.*, pages 279–286, 1998.

[20] T. M. Chan. Dynamic planar convex hull operations in near-logarithmic amortized time. To appear in *Proc. 40th IEEE Sympos. Found. Comput. Sci.*, 1999.

[21] T. M. Chan. Remarks on k-level algorithms in the plane. Manuscript, 1999.

[22] B. Chazelle. On the convex layers of a planar set. *IEEE Trans. Inform. Theory*, IT-31:509–517, 1985.

[23] B. Chazelle. Filtering search: a new approach to query-answering. *SIAM J. Comput.*, 15:703–724, 1986.

[24] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete Comput. Geom.*, 10:377–409, 1993.

[25] B. Chazelle, R. Cole, F. P. Preparata, and C. K. Yap. New upper bounds for neighbor searching. *Inform. Control*, 68:105–124, 1986.

[26] B. Chazelle and H. Edelsbrunner. An improved algorithm for constructing kth-order Voronoi diagrams. *IEEE Trans. Comput.*, C-36:1349–1354, 1987.

[27] B. Chazelle and J. Friedman. A deterministic view of random sampling and its use in geometry. *Combinatorica*, 10:229–249, 1990.

[28] B. Chazelle, L. Guibas, and D. T. Lee. The power of geometric duality. *BIT*, 25:76–90, 1985.

[29] B. Chazelle and F. P. Preparata. Halfspace range search: an algorithmic application of k-sets. *Discrete Comput. Geom.*, 1:3–93, 1986.

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

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

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

[33] R. Cole, M. Sharir, and C. K. Yap. On k-hulls and related problems. *SIAM J. Comput.*, 16:61-77, 1987.

[34] T. K. Dey. Improved bounds on planar k-sets and k-levels. *Discrete Comput. Geom.*, 19:373–382, 1998.

[35] T. K. Dey and H. Edelsbrunner. Counting triangle crossings and halving planes. *Discrete Comput. Geom.*, 12:281–289, 1994.

[36] H. Edelsbrunner. Edge-skeletons in arrangements with applications. *Algorithmica*, 1:93–109, 1986.

[37] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag, Berlin, 1987.

[38] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Trans. on Graphics*, 9:66–104, 1990.

[39] H. Edelsbrunner, J. O'Rourke, and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. *SIAM J. Comput.*, 15:341–363, 1986.

[40] H. Edelsbrunner and E. Welzl. Constructing belts in two-dimensional arrangements with applications. *SIAM J. Comput.*, 15:271–284, 1986.

[41] P. Erdős, L. Lovász, A. Simmons, and E. Straus. Dissection graphs of planar point sets. In *A Survey of Combinatorial Theory* (J. N. Srivastava, ed.), North-Holland, Amsterdam, Netherlands, pages 139–154, 1973.

[42] H. Everett, J.-M. Robert, and M. van Kreveld. An optimal algorithm for the ($\leq k$)-levels, with applications to separation and transversal problems. *Int. J. Comput. Geom. Appl.*, 6:247–261, 1996.

[43] D. Halperin. Arrangements. In *Handbook of Discrete and Computational Geometry* (J. E. Goodman and J. O'Rourke, ed.), pages 389–412, CRC Press, 1997.

[44] S. Har-Peled. Taking a walk in a planar arrangement. To appear in *Proc. 40th IEEE Sympos. Found. Comput. Sci.*, 1999.

[45] N. Katoh and K. Iwano. Finding k farthest pairs and k closest/farthest bichromatic pairs for points in the plane. *Int. J. Comput. Geom. Appl.*, 5:37–51, 1995.

[46] D. T. Lee. On k-nearest neighbor Voronoi diagrams in the plane. *IEEE Trans. Comput.*, C-31:478–287, 1982.

[47] J. Matoušek. Efficient partition trees. *Discrete Comput. Geom.*, 8:315–334, 1992.

[48] J. Matoušek. Reporting points in halfspaces. *Comput. Geom. Theory Appl.*, 2:169–186, 1992.

[49] J. Matoušek. Geometric range searching. *ACM Comput. Surv.*, 26:421–461, 1994.

[50] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, New York, 1995.

[51] K. Mulmuley. Output sensitive construction of levels and Voronoi diagrams in R^d of order 1 to k. In *Proc. 22nd ACM Sympos. Theory Comput.*, pages 322–330, 1990.

[52] K. Mulmuley. On levels in arrangements and Voronoi diagrams. *Discrete Comput. Geom.*, 6:307–338, 1991.

[53] K. Mulmuley. *Computational Geometry: An Introduction Through Randomized Algorithms.* Prentice-Hall, Englewood Cliffs, N.J., 1994.

[54] F. P. Preparata and S. J. Hong. Convex hulls of finite sets of points in two and three dimensions. *Commun. ACM*, 20:87–93, 1977.

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

[56] E. Ramos. On range reporting, ray shooting, and k-level construction. In *Proc. 15th ACM Sympos. Comput. Geom.*, pages 390–399, 1999.

[57] M. I. Shamos and D. Hoey. Closest-point problems. In *Proc. 16th IEEE Sympos. Found. Comput. Sci.*, pages 151–162, 1977.