SIAM J. COMPUT.
Vol. 26, No. 6, pp. 1714–1732, December 1997

008

COMPUTING ENVELOPES IN FOUR DIMENSIONS WITH APPLICATIONS*

PANKAJ K. AGARWAL†, BORIS ARONOV‡, AND MICHA SHARIR§

Abstract. Let $\mathcal{F}$ be a collection of n d-variate, possibly partially defined, functions, all algebraic of some constant maximum degree. We present a randomized algorithm that computes the vertices, edges, and 2-faces of the lower envelope (i.e., pointwise minimum) of $\mathcal{F}$ in expected time $O(n^{d+\varepsilon})$ for any $\varepsilon > 0$. For $d = 3$, by combining this algorithm with the point-location technique of Preparata and Tamassia, we can compute, in randomized expected time $O(n^{3+\varepsilon})$, for any $\varepsilon > 0$, a data structure of size $O(n^{3+\varepsilon})$ that, for any query point q, can determine in $O(\log^2 n)$ time the function(s) of $\mathcal{F}$ that attain the lower envelope at q. As a consequence, we obtain improved algorithmic solutions to several problems in computational geometry, including (a) computing the width of a point set in 3-space, (b) computing the "biggest stick" in a simple polygon in the plane, and (c) computing the smallest-width annulus covering a planar point set. The solutions to these problems run in randomized expected time $O(n^{17/11+\varepsilon})$, for any $\varepsilon > 0$, improving previous solutions that run in time $O(n^{8/5+\varepsilon})$. We also present data structures for (i) performing nearest-neighbor and related queries for fairly general collections of objects in 3-space and for collections of moving objects in the plane and (ii) performing ray-shooting and related queries among n spheres or more general objects in 3-space. Both of these data structures require $O(n^{3+\varepsilon})$ storage and preprocessing time, for any $\varepsilon > 0$, and support polylogarithmic-time queries. These structures improve previous solutions to these problems.

Key words. lower envelopes, point location, ray shooting, closest pair

AMS subject classifications. 68Q20, 68Q25, 68R05, 68U05

PII. S0097539794265724

1. Introduction.

Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be a collection of n d-variate, possibly partially defined, functions, all algebraic of some constant maximum degree b (and if they are partially defined, their domains of definition are also described each by a constant number of polynomial equalities and inequalities of maximum degree b). Abusing the notation slightly, we will not distinguish between a function and its graph. The *lower envelope* $E_{\mathcal{F}}$ of $\mathcal{F}$ is defined as

$$E_{\mathcal{F}}(\mathbf{x}) = \min_i f_i(\mathbf{x}),$$

where the minimum is taken over all functions of $\mathcal{F}$ that are defined at $\mathbf{x}$. The *minimization diagram* $M_{\mathcal{F}}$ of $\mathcal{F}$ is the decomposition of $\mathbb{R}^d$ into maximal connected regions (of any dimension), called *cells* (or *faces*), so that within each cell the same subset of functions appears on the envelope $E_{\mathcal{F}}$. There is a natural subdivision of

* Received by the editors April 6, 1994; accepted for publication (in revised form) November 27, 1995. Work on this paper by the first author was supported by NSF grant CCR-93–01259 and by an NYI award. Work on this paper by the second author was supported by NSF grant CCR-92-11541. Work by the third author was supported by NSF grant CCR-91-22103, by a Max Planck Research Award, and by grants from the U.S.–Israeli Binational Science Foundation, the German–Israeli Foundation for Scientific Research and Development, and the Fund for Basic Research administered by the Israeli Academy of Sciences.

http://www.siam.org/journals/sicomp/26-6/26572.html

† Department of Computer Science, Box 90129, Duke University, Durham, NC 27708-0129 (pankaj@euclid.cs.duke.edu).

‡ Department of Computer and Information Science, Polytechnic University, Brooklyn, NY 11201-3840 (aronov@ziggy.poly.edu).

§ School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel (sharir@math.tau.ac.il) and Courant Institute of Mathematical Sciences, New York University, New York, NY 10012.

$E_{\mathcal{F}}$ into cells, as well, where a cell of the lower envelope is defined as the portion $E_{\mathcal{F}}$ that projects onto cell of $M_{\mathcal{F}}$. As the correspondence is 1-1, we will abuse the terminology and not make a distinction between the two kinds of cells. (A more detailed definition, treating also the case of partially defined functions, is given in [Sha].) The *combinatorial complexity* of $M_{\mathcal{F}}$ and of $E_{\mathcal{F}}$ is the number of faces of all dimensions in $M_{\mathcal{F}}$ and $E_{\mathcal{F}}$.

Recently, there has been a significant progress in the analysis of the combinatorial complexity of lower envelopes of multivariate functions [HS, Sha]. In particular, it was shown in [Sha] that the maximum complexity of $M_{\mathcal{F}}$ is $O(n^{d+\varepsilon})$, for any $\varepsilon > 0$, where the constant of proportionality depends on ε, d, and b. This result almost settles a major open problem and has already led to many applications [ASa, HS, Sha]. However, less progress has been made on the corresponding algorithmic problem, which calls for the efficient construction of the lower envelope of such a collection $\mathcal{F}$, in time $O(n^{d+\varepsilon})$, for any $\varepsilon > 0$. The ideal output of an algorithm that computes the envelope is a data structure of size $O(n^{d+\varepsilon})$, which can return, for a given point $\mathbf{x} \in \mathbb{R}^d$, the identity of the function(s) attaining the envelope at $\mathbf{x}$ in logarithmic (or polylogarithmic) time. Weaker solutions might provide just an enumeration of the cells of $M_{\mathcal{F}}$ and adjacency structure representing all pairs of cells that touch each other.

Sharir [Sha] presented an algorithm for computing the lower envelope of bivariate functions having the properties listed above, which runs in time $O(n^{2+\varepsilon})$, for any $\varepsilon > 0$. Other algorithms with a similar performance are given in [BD, dBDS]. The simplest solution to this problem, involving a deterministic divide-and-conquer algorithm, was recently presented in [ASS]. All these algorithms facilitate point-location queries of the sort described above. Unfortunately, these methods fail in higher dimensions. For example, the technique of [Sha] relies on the existence of a *vertical decomposition* of the minimization diagram M_R of a sample R of r functions of $\mathcal{F}$, into a small number (that is, $O(r^{d+\varepsilon})$) of cells of *constant description complexity*. Such decompositions exist (and are easy to compute) for $d = 2$, but their existence in higher dimensions is still an open problem.

In this paper we present a randomized algorithm with $O(n^{d+\varepsilon})$ expected time, for any $\varepsilon > 0$, for computing the vertices, edges, and 2-dimensional faces of the lower envelope of n d-variate functions having the properties assumed above. We can use this algorithm to compute the entire lower envelope of a collection $\mathcal{F}$ of n trivariate functions, which has all the desired characteristics; in particular, it preprocesses $\mathcal{F}$, in expected time $O(n^{3+\varepsilon})$, into a data structure of size $O(n^{3+\varepsilon})$ that, for a query point q, can compute $E_{\mathcal{F}}(q)$, and also the function(s) attaining the envelope at q, in $O(\log^2 n)$ time. The algorithm bypasses the problem of having to construct small-size vertical decomposition by applying the technique of Preparata and Tamassia [PT] for point location in certain types of 3-dimensional subdivisions. This allows us to use a coarser decomposition of the minimization diagram, whose size is close to cubic.

Several recent papers [AST, CEGSb, MS] have studied a variety of geometric problems whose solution calls for the construction of, and searching in, lower or upper envelopes in 4-space. These applications fall into two main categories: *preprocess-for-queries* problems, which call for the construction of such an envelope, to be queried repeatedly later; and *batched* problems, where all the queries are known in advance and the goal is to produce the answers to all of them efficiently. We will present some of these applications, as listed in the abstract, where our new algorithm for computing lower envelopes in four dimensions leads to improved solutions.

The paper is organized as follows. In section 2 we present a general efficient randomized technique, which we believe to be of independent interest, for computing all the 0, 1, and 2-dimensional features of lower envelopes in any dimension. Then, in section 3, we apply this algorithm to the case of trivariate functions. This gives us an initial representation of the lower envelope of such a collection, which we then augment by additional features, resulting in a representation suitable for the application of the Preparata–Tamassia technique. In sections 4 and 5 we present applications of our result to a variety of problems in computational geometry, as listed in the abstract.

After the original submission of this paper, Agarwal and Sharir [ASb] observed that, for some of the problems that are solved here, the size of the vertical decomposition of the relevant minimization diagrams is small. This leads to slightly improved solutions to these problems. See also the remark at the end of section 5.1.

2. Lower envelopes in arbitrary dimension. In this section, we present a randomized technique for computing a partial description of lower envelopes of d-variate functions, for $d \geq 2$, whose expected running time is $O(n^{d+\varepsilon})$, for any $\varepsilon > 0$. This technique only computes the vertices, edges, and 2-faces of the envelope, which is sufficient for the full construction of envelopes in 4-space, as will be explained in section 3.

Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be a collection of (partial) d-variate functions in $\mathbb{R}^{d+1}$, satisfying the conditions described in the introduction. We assume that the functions of $\mathcal{F}$ are in general position; see [Sha] for more details and for a discussion of this assumption. To compute the lower envelope $E_{\mathcal{F}}$ of $\mathcal{F}$, in the above partial sense, we proceed as follows. Let Σ be the family of all $(d-1)$-subsets of $\mathcal{F}$. We fix a subset $\sigma = \{f_1, \ldots, f_{d-1}\} \in \Sigma$, and let

$$\Pi^\sigma = \{\mathbf{x} \in \mathbb{R}^d \mid f_1(\mathbf{x}) = \cdots = f_{d-1}(\mathbf{x})\}.$$

Since the f_i's are assumed to be in general position, Π^σ is a 2-dimensional surface (or a surface patch). For the sake of simplicity, we assume that Π^σ is connected and x_1x_2-monotone (i.e., any $(d-2)$-flat orthogonal to the x_1x_2-plane meets Π^σ in at most one point); otherwise, we decompose it into a constant number of connected portions so that each portion is an x_1x_2-monotone surface patch, and work with each patch separately. For each $i \geq d$, let

$$\gamma_i = \{\mathbf{x} \in \Pi^\sigma \mid f_i(\mathbf{x}) = f_1(\mathbf{x}) = \cdots = f_{d-1}(\mathbf{x})\};$$

γ_i is a 1-dimensional curve, which partitions Π^σ into two (not necessarily connected) regions, K_i^+ and K_i^-, where

$$K_i^+ = \{\mathbf{x} \in \Pi^\sigma \mid f_i(\mathbf{x}) > f_1(\mathbf{x}) = \cdots = f_{d-1}(\mathbf{x})\},$$
$$K_i^- = \{\mathbf{x} \in \Pi^\sigma \mid f_i(\mathbf{x}) < f_1(\mathbf{x}) = \cdots = f_{d-1}(\mathbf{x})\}.$$

Then the intersection $Q^\sigma = \bigcap_{i \geq d} K_i^+$ is the portion of Π^σ over which the envelope $E_{\mathcal{F}}$ is attained by the functions of σ. The algorithm will compute the regions Q^σ, over all choices of $(d-1)$-tuples σ of functions, thereby yielding the vertices, edges, and 2-faces of $M_{\mathcal{F}}$ (because of the general position assumption, any such feature must show up as a feature of at least one of the regions Q^σ). The algorithm actually computes the vertices, edges, and 2-faces of a refinement of $M_{\mathcal{F}}$, but the faces of $M_{\mathcal{F}}$ of dimension at most 2 can be retrieved from them in a straightforward manner. We omit the easy details.

We compute Q^σ using a randomized incremental approach, similar to the ones described in [CEGSS, dBDS, Mua, Mub, SA]. Since the basic idea is by now fairly standard, we give only a brief overview of the algorithm and refer the reader to [CEGSS, SA] for details. We first compute the set $\Gamma^\sigma = \{\gamma_i \mid d \le i \le n\}$.[1] Next, we add the curves γ_i one by one in a random order and maintain the intersection of the regions K_i^+ for the curves added so far. Let $(\gamma_d, \gamma_{d+1}, \ldots, \gamma_n)$ be the (random) insertion sequence, and let Q_i^σ denote the intersection of $K_d^+, \ldots, K_i^+$. We construct and maintain the "vertical decomposition" $\tilde{Q}_i^\sigma$ of Q_i^σ. This is defined as the partitioning of each 2-face ϕ of Q_i^σ into "pseudotrapezoids," obtained by drawing, from each vertex of ϕ and from each locally x_1-extreme point on $\partial\phi$, a curve within ϕ obtained by intersecting ϕ with the hyperplane $x_1 = const$, and by extending it (in both directions if necessary) until it meets $\partial\phi$ again. Each pseudotrapezoid is *defined* by at most four curves of Γ^σ; conversely, any four or fewer curves of Γ^σ define a constant number of pseudotrapezoidal cells, namely, those formed along Π^σ when only these curves are inserted. (Note that this construction is well defined since Π^σ is an x_1x_2-monotone surface.) In the $(i+1)$st step we add K_{i+1}^+ and compute $\tilde{Q}_{i+1}^\sigma$ from $\tilde{Q}_i^\sigma$, using the technique described in [CEGSS].

The analysis of the expected running time of the algorithm proceeds along the same lines as described in [CEGSS, SA]. We define the *weight*, $w(\tau)$, of a pseudotrapezoid τ, defined by the arcs of Γ^σ, to be the number of functions f_i, for $i = d, d+1, \ldots, n$, excluding the up to four functions whose intersections with Π^σ *define* τ, such that $f_i(\mathbf{x}) < f_1(\mathbf{x}) = \cdots = f_{d-1}(\mathbf{x})$ for some point $\mathbf{x} \in \tau$.

As shown in [CEGSS], the cost of the above procedure is proportional to the number of pseudotrapezoids that are created during the execution of the algorithm, plus the sum of their weights, plus an overhead term of $O(n^d)$ needed to prepare the collections of curves γ_i over all 2-dimensional intersection manifolds Π^σ. The analysis given below deals only with pseudotrapezoids that are defined by exactly four such functions (plus the $d-1$ functions defining Π^σ), and easy and obvious modifications are necessary to handle all other pseudotrapezoids.

Let T^σ denote the set of pseudotrapezoids (or "cells" for brevity) defined by four arcs of Γ^σ, and let $T = \bigcup_{\sigma\in\Sigma} T^\sigma$. Each cell in T is defined by $d-1+4 = d+3$ functions of $\mathcal{F}$. In what follows we implicitly assume that the specification of a cell $\tau \in T$ includes the $d+3$ functions defining τ, where there is a clear distinction between the first $d-1$ functions (constituting the set σ) and the last four functions (defining the four curves that generate τ along Π^σ). For an integer $k \ge 0$ and a subset $R \subseteq \mathcal{F}$, let $T_k(R) \subseteq T$ (resp., $T_{\le k}(R) \subseteq T$) denote the set of cells, defined by $d+3$ functions of R, as above, with weight k (resp., at most k). Let $N_k(R) = |T_k(R)|$ and

$$N_k(r) = \max_R N_k(R)\,,$$

where the maximum is taken over all subsets R of $\mathcal{F}$ of size r. Similarly, we define $N_{\le k}(R)$ and $N_{\le k}(r)$. Since each cell of $T_0(R)$ lies on the lower envelope of R, it follows that $N_0(r) = O(r^{d+\varepsilon})$. Adapting the analysis technique of Clarkson and Shor [ClS], we have the following lemma.

[1] We need to assume an appropriate model of computation, in which any of the various primitive operations required by the algorithm can be performed in constant time. For example, we can assume the model used in real computational algebraic geometry [HRR], where each algebraic operation involving a constant number of polynomials of constant maximum degree can be performed exactly, using rational arithmetic in constant time.

LEMMA 2.1. *The probability that a pseudotrapezoidal cell $\tau \in T_k(\mathcal{F})$ is created during the incremental construction of Q^σ, where $\sigma \in \Sigma$ is the tuple for which Π^σ contains τ, is $1/\binom{k+4}{4}$.*

Proof. For τ to be created, it is necessary and sufficient that the four curves of Γ^σ defining τ appear in the random insertion order before any of the curves corresponding to the k functions that contribute to the weight of τ, and this probability is easily seen to be $1/\binom{k+4}{4}$. □

LEMMA 2.2. *For any $0 \le k \le n-d-3$ and for any $\varepsilon > 0$,*

$$N_{\le k}(n) = O((k+1)^{3-\varepsilon} n^{d+\varepsilon}).$$

Proof. We use a variant of the probabilistic analysis technique of Clarkson and Shor [ClS]. If we choose a random sample $R \subseteq \Gamma$ of $r = \lfloor n/(k+1) \rfloor$ functions of $\mathcal{F}$, then a cell $\tau \in T_k(\Gamma)$ is in $T_0(R)$ if all $d+3$ functions, $f_1, \ldots, f_{d+3}$, defining τ are chosen in R, and none of the remaining k functions f_i, such that $f_i(\mathbf{x}) \le f_1(\mathbf{x})$ for some $\mathbf{x} \in \tau$, are chosen in R. The Clarkson–Shor technique implies that

$$\begin{aligned} N_{\le k}(n) &= O((k+1)^{d+3} N_0(\lfloor n/(k+1) \rfloor)) \\ &= O((k+1)^{d+3} (n/(k+1))^{d+\varepsilon}) \\ &= O((k+1)^{3-\varepsilon} n^{d+\varepsilon}), \end{aligned}$$

for any $\varepsilon > 0$. □

For a cell $\tau \in T$, let A_τ be the event that $\tau \in \tilde{Q}_i^\sigma$, for the tuple $\sigma \in \Sigma$ for which T^σ contains τ, and for some $d \le i \le n$. The expected running time of the algorithm, over all choices of $(d-1)$-tuples of functions, is thus proportional to

$$\begin{aligned} \sum_{\tau \in T} \Big[(w(\tau)+1) \cdot \Pr[A_\tau] \Big] + O(n^d) &= \sum_{k \ge 0} \sum_{\tau \in T_k(\mathcal{F})} \Big[(k+1) \cdot \Pr[A_\tau] \Big] + O(n^d) \\ &= \sum_{k=0}^{n-d-3} \frac{(k+1) N_k(\mathcal{F})}{\binom{k+4}{4}} + O(n^d), \end{aligned}$$

where the last inequality follows from Lemma 2.1. Since

$$N_k(\mathcal{F}) = N_{\le k}(\mathcal{F}) - N_{\le (k-1)}(\mathcal{F}),$$

we obtain, using Lemma 2.2,

$$\begin{aligned} &\sum_{k=0}^{n-d-3} \frac{(k+1) N_k(\mathcal{F})}{\binom{k+4}{4}} \\ &= N_0(\mathcal{F}) + 24 \sum_{k=1}^{n-d-3} \frac{N_{\le k}(\mathcal{F}) - N_{\le (k-1)}(\mathcal{F})}{(k+4)(k+3)(k+2)} \\ &= O(n^{d+\varepsilon}) + 24 \sum_{k=1}^{n-d-4} N_{\le k}(n) \left[\frac{1}{(k+4)(k+3)(k+2)} - \frac{1}{(k+5)(k+4)(k+3)} \right] \\ &\qquad + \frac{24 N_{\le n-d-3}(n)}{(n-d+1)(n-d)(n-d-1)} \\ &= O\left(n^{d+\varepsilon} + \sum_{k=1}^{n-d-4} \frac{k^{3-\varepsilon} n^{d+\varepsilon}}{(k+5)(k+4)(k+3)(k+2)} \right) \qquad \text{(using Lemma 2.2)} \end{aligned}$$

$$= O\left(n^{d+\varepsilon} \cdot \sum_{k=1}^{n-d-4} \frac{1}{k^{1+\varepsilon}}\right) = O(n^{d+\varepsilon}) .$$

We thus obtain the following result.

THEOREM 2.3. *The vertices, edges, and 2-faces of the lower envelope of n (partial) d-variate functions, satisfying the conditions stated in the introduction, can be computed in randomized expected time $O(n^{d+\varepsilon})$, for any $\varepsilon > 0$.*

Remark. The preceding analysis shows that, for $d \geq 2$, the expected running time of our algorithm is in fact $O(n^d + \sum_{k=1}^{n} k^{d-1}\varphi(\lfloor n/k \rfloor))$, where $\varphi(r)$ is an upper bound on the complexity of the lower envelope of any subset of S of size at most r. For example, the vertices, edges, and 2-faces of the lower envelope of n simplices in $\mathbb{R}^{d+1}$ can be computed in $O(n^d \alpha(n) \log n)$ expected time, because the complexity of the lower envelope of n d-simplices is $O(n^d \alpha(n))$ [PS].

3. Envelopes in four dimensions. We next apply the results of the preceding section to obtain an efficient algorithm for constructing the lower envelope of a collection $\mathcal{F}$ of n trivariate functions, satisfying the assumptions made above in the following strong sense: One can preprocess $\mathcal{F}$ in randomized expected time $O(n^{3+\varepsilon})$, for any $\varepsilon > 0$, into a data structure of size $O(n^{3+\varepsilon})$ that supports queries of the form: given a point $w \in \mathbb{R}^3$, compute the function(s) attaining $E_{\mathcal{F}}$ at w; each query can be answered in $O(\log^2 n)$ time.

To achieve this we first apply the algorithm summarized in Theorem 2.3 to compute the vertices, edges, and 2-faces of the minimization diagram $M_{\mathcal{F}}$ of $\mathcal{F}$. We next partition each cell of $M_{\mathcal{F}}$ into "monotone" subcells, in the sense of Lemma 3.1, to obtain a refinement $M'_{\mathcal{F}}$ of $M_{\mathcal{F}}$. This refinement satisfies the requirements of the point-location method due to Preparata and Tamassia [PT], which we use to produce the data structure representing the lower envelope in the above manner.

We define and construct $M'_{\mathcal{F}}$ as follows. We mark, along each 2-face F of $M_{\mathcal{F}}$, the locus γ_F of all points of F which are either singular or have a vertical tangency (in the z-direction). The arcs γ_F, for all 2-faces F, lie along $O(n^2)$ curves in $\mathbb{R}^3$, each being the xyz-projection of (i) the locus of all singular points or points with z-vertical tangency along some 2-manifold $f_i = f_j$, for a pair of indices $i \neq j$, or (ii) of points along the boundary of some f_i. We consider below only the former case; the latter case can be handled in almost the same (and, actually, simpler) manner. Let δ be one of these curves. We consider the 2-dimensional surface V_δ, within the xyz-space, obtained as the union of all lines passing through points of δ and parallel to the z-axis; let V_δ^+, V_δ^- denote the portions of V_δ that lie, respectively, above and below δ. (Since δ is not necessarily an xy-monotone arc, V_δ may have self-intersections along some vertical lines, along which V_δ^+ and V_δ^- are not well defined; we omit here details of the (rather easy) handling of such cases.)

Let δ_0 be the portion of δ over which the functions f_i and f_j attain the envelope $E_{\mathcal{F}}$. Clearly, δ_0 is the union of all arcs γ_F that are contained in δ, and the number of connected components in δ_0, summed over all intersection curves δ, is $O(n^{3+\varepsilon})$. Let w be a point in δ_0. Then the cell c of $M_{\mathcal{F}}$ lying immediately above w in the z-direction is such that within c the envelope $E_{\mathcal{F}}$ is attained by either f_i or f_j. Thus the upward-directed z-vertical ray emanating from w leaves c (if at all) at a point above w that lies on the xyz-projection of a 2-manifold of the form $f_i = f_k$ or $f_j = f_k$. (In addition, it is also possible that the ray will encounter a point on the projection of the boundary of the domain of f_i or f_j—this can be handled by similar, but easier, means.) For fixed i and j, there are only $O(n)$ possible 2-manifolds of this kind (i.e., xyz-projections

of surfaces of the form $f_i = f_k$ or $f_j = f_k$). We compute the lower envelope $E^{(\delta)}$ of these $O(n)$ 2-manifolds, restricted to V_δ^+. It is easily seen that the complexity of $E^{(\delta)}$ is $O(\lambda_q(n))$, for some constant q depending on the maximum degree of these 2-manifolds; $\lambda_q(n)$ is the maximum length of Davenport–Schinzel sequences of order q that are composed of n symbols and is close to linear in n for any fixed q [ASS, SA]. We next take the portions of the graph of $E^{(\delta)}$ that lie over δ_0 and "etch" them along the corresponding 2-faces of $M_\mathcal{F}$. We apply a fully symmetric procedure within V_δ^- and repeat these steps for all curves δ. The overall combinatorial complexity of all the added curves is thus $O(n^2\lambda_q(n) + n^{3+\varepsilon}) = O(n^{3+\varepsilon})$, for any $\varepsilon > 0$, and they can be computed in $O(n^{3+\varepsilon})$ time, as is easily verified.

Let $M'_\mathcal{F}$ denote the refined cell decomposition of $\mathbb{R}^3$, obtained by adding to $M_\mathcal{F}$, for each of the curves δ, the arcs γ_F, the etched arcs of the upper and lower envelopes in the vertical manifolds V_δ, and the z-vertical walls (i.e., union of all z-vertical segments) contained in V_δ and connecting the arcs γ_F to the etched arcs. If an edge of $M'_\mathcal{F}$ is not monotone in the y-direction, we split it into $O(1)$ edges by adding a vertex at every local y-extremal point on this edge. As just argued, the combinatorial complexity of $M'_\mathcal{F}$ is still $O(n^{3+\varepsilon})$, and $M'_\mathcal{F}$ has the following crucial property.

LEMMA 3.1. *For each 3-cell c of $M'_\mathcal{F}$, every connected component of a cross section of c by a plane parallel to the xz-plane is x-monotone.*

Proof. Suppose the contrary, and let π be a plane parallel to the xz-plane, for which there exists a connected component c' of $c \cap \pi$ that is not x-monotone. Then there is a point $w \in \partial c'$ so that the z-vertical line passing through w meets c' both slightly above and slightly below w. But then w is either a singular point or a point with z-vertical tangency lying on one of the curves γ_F. By construction, $M'_\mathcal{F}$ must contain the vertical segment passing through w and contained in c, as part of some vertical wall, a contradiction. This completes the proof of the lemma. □

Let $M'_\mathcal{F}(y_0)$ denote the cross section of $M'_\mathcal{F}$ by the plane $y = y_0$. Lemma 3.1 implies that $M'_\mathcal{F}(y_0)$ is an x-monotone planar subdivision, for each y_0. Hence, if we orient the edges of $M'_\mathcal{F}$ in the positive x-direction, add a pair of nominal points s, t at $x = -\infty$ and $x = +\infty$, respectively, and connect them to the appropriate unbounded edges of $M'_\mathcal{F}(y_0)$, this map becomes a planar st-graph, in the notation of Preparata and Tamassia [PT]. (Note that the vertical decomposition that generates $M'_\mathcal{F}$ from $M_\mathcal{F}$ may create z-vertical edges; if these edges are oriented consistently, say in the upward z-direction, then $M'_\mathcal{F}$ remains an st-graph.) We denote this st-graph also by $M'_\mathcal{F}(y_0)$.

LEMMA 3.2. *Let I be an open interval of the y-axis that does not contain the y-coordinate of any vertex of $M'_\mathcal{F}$. Then, for each $y_1, y_2 \in I$, $M'_\mathcal{F}(y_1)$, $M'_\mathcal{F}(y_2)$ are isomorphic, as labeled embedded planar st-graphs.*

Proof. We call a y-coordinate *critical* if it is the y-coordinate of a vertex of $M'_\mathcal{F}$. Since each vertex of $M'_\mathcal{F}(\cdot)$ is an intersection of the sweep plane with an edge of $M'_\mathcal{F}$ and since each edge of $M'_\mathcal{F}$ is y-monotone, the set of vertices of $M'_\mathcal{F}(y)$ can change only at critical values of y. In other words, the set of vertices of the cross section $M'_\mathcal{F}(\cdot)$ does not change combinatorially between any two consecutive critical values. Let (u_1, v_1) be a nonvertical directed edge of $M'_\mathcal{F}(y_1)$, and let u_2, v_2 be the vertices of $M'_\mathcal{F}(y_2)$ corresponding, respectively, to u_1 and v_1 (i.e., the intersections of the sweep plane with the same edges of $M'_\mathcal{F}$). Suppose to the contrary that (u_2, v_2) is not an edge of $M'_\mathcal{F}(y_2)$. There are several ways in which the edge (u_1, v_1) can disappear during the sweep:

(i) it might cease to be x-monotone (with a vertical inflection point, or another

singular point, appearing in its middle),
(ii) some vertical edge might appear and split it in two,
(iii) its two vertices might approach each other and merge into a common vertex, or
(iv) either of the vertices might split into two vertices.

However, as is easily verified, the y-coordinate of any of these events must be one of the critical values of y, contrary to assumption that the interval I does not contain the y-coordinate of any vertex of $M'_{\mathcal{F}}$. A similar argument handles the case where (u_1, v_1) is a vertical edge of $M'_{\mathcal{F}}(y_1)$: such an edge is erected if one of these vertices, say u_1, has a z-vertical tangency, or is otherwise singular along one of the surfaces of $M_{\mathcal{F}}$. As we sweep from y_1 to y_2, this vertical edge might disappear if any of the following events occurs:

(i) the point corresponding to u_1 changes its singular status (for simplicity of exposition, the term "singular" refers here both to singular points and to points with z-vertical tangency);
(ii) the point corresponding to u_1 splits into two vertices or merges with another vertex of the cross section;
(iii) the point corresponding to v_1 splits into two vertices or merges with another vertex;
(iv) the point corresponding to v_1 becomes singular on its surface; or
(v) another singular point appears on the vertical segment u_1v_1.

Again, for any of these events, its y-coordinate must be critical, contrary to assumption. This completes the proof of the lemma. □

We are now in a position to apply the point-location technique of Preparata and Tamassia [PT]. They show that a 3-dimensional subdivision of combinatorial complexity N can be preprocessed in time $O(N \log^2 N)$ into a data structure of size $O(N \log^2 N)$, which supports $O(\log^2 N)$-time point-location queries in the given subdivision. However, for this to hold, the subdivision must have the following two properties:

(a) the 1-skeleton of each cross section of the subdivision by a plane parallel to the xz-plane is a planar st-graph, whose faces are all monotone in the x-direction and whose edges are all oriented in the positive x-direction (or, for vertical edges, in the positive z-direction).
(b) At each point where the cross section, as an st-graph, changes, the graph can be updated by a constant number of operations from the following collection:
  (b.1) insertion of a vertex in the middle of an edge, or the complementary operation of deleting a vertex of in-degree and out-degree 1 and replacing the two incident edges by one, while maintaining x-monotonicity of the incident faces;
  (b.2) insertion of an edge partitioning an x-monotone face into two x-monotone subfaces, or, conversely, deletion of an edge and merging its two adjacent faces, provided their union is x-monotone;
  (b.3) merging two adjacent vertices into one vertex (collapsing the edge connecting them) or splitting a vertex into two vertices (forming a new edge between these vertices), again maintaining x-monotonicity.

It is easily verified that, indeed, each change occurring in the structure of the cross section $M'_{\mathcal{F}}(\cdot)$, at any of the critical y values in Lemma 3.2, can be expressed as a constant number of operations of the types mentioned above. In summary, we thus obtain the following main result of the paper.

THEOREM 3.3. *Let $\mathcal{F}$ be a given collection of n trivariate, possibly partially defined, functions, all algebraic of constant maximum degree, and whose domains of definition (if they are partially defined) are each defined by a constant number of algebraic equalities and inequalities of constant maximum degree. Then, for any $\varepsilon > 0$, the lower envelope $E_{\mathcal{F}}$ of $\mathcal{F}$ can be computed in randomized expected time $O(n^{3+\varepsilon})$, and stored in a data structure of size $O(n^{3+\varepsilon})$, so that, given any query point $w \in \mathbb{R}^3$, we can compute $E_{\mathcal{F}}(w)$, as well as the function(s) attaining $E_{\mathcal{F}}$ at w, in $O(\log^2 n)$ time.*

4. Applications: Query problems. In this section we apply Theorem 3.3 to two problems involving preprocessing and querying certain collections of objects in 3-space.

4.1. Ray shooting amidst spheres. Given a collection $\mathcal{S}$ of n spheres in 3-space, we wish to preprocess $\mathcal{S}$ into a data structure that supports *ray-shooting* queries, each seeking the first sphere, if any, met by a query ray. This problem has recently been studied in [AMb, AGPS, MS]. The first two papers present a data structure that requires $O(n^{4+\varepsilon})$ storage and preprocessing, for any $\varepsilon > 0$, and answers a query in time $O(\log^2 n)$. The third paper [MS] gives a rather elaborate and improved solution that requires $O(n^{3+\varepsilon})$ preprocessing and storage, for any $\varepsilon > 0$, and answers a query in $O(n^{\varepsilon})$ time.

These algorithms use the technique described in [AMa] and reduce the problem to that of determining whether a query segment e intersects any sphere of $\mathcal{S}$. Then, using a multilevel data structure, they reduce the problem to that of detecting whether the line λ containing the segment e intersects any sphere of $\mathcal{S}$. As observed in [AGPS, MS], this problem can be further reduced to point location in the upper envelope of a set of certain trivariate functions, as follows. Let π be the plane passing through λ and orthogonal to the vertical plane V passing through λ. Let π^+ (resp., π^-) be the half-space lying above (resp., below) π. Let S be a sphere whose center lies in π^+ and that intersects V in a disc D. Then the center of D lies above λ, so either λ intersects S or passes below S, in the sense that λ and S are disjoint and there is a point on λ that lies vertically below a point in S (see Figure 1). A similar property holds if S intersects V and its center lies in π^-. We preprocess the centers of spheres of $\mathcal{S}$ into a half-space range-searching data structure of size $O(n^{3+\varepsilon})$. Then, for a query λ, we can decompose $\mathcal{S}$ into $O(1)$ canonical subsets, so that, within each subset, either the centers of all spheres lie in π^+ or they all lie in π^-. Let us consider the case when the centers of all spheres lie in π^+.

Hence, we need to solve the following subproblem: Given a set $\mathcal{S}$ of n spheres in 3-space, and given a query line λ with the property that for each sphere $S \in \mathcal{S}$, either λ intersects S or there is no point of S lying vertically below λ, determine whether λ intersects any sphere of $\mathcal{S}$. We reduce this problem to point location in the lower envelope of certain trivariate functions as follows. We can parametrize a line λ in 3-space by four parameters $(\xi_1, \xi_2, \xi_3, \xi_4)$, so that the equations defining λ are $y = x\xi_1 + \xi_2$, $z = x\xi_3 + \xi_4$. (We assume here that λ is not parallel to the yz-plane; such lines can be handled in a different, and much simpler manner.) For each sphere $S \in \mathcal{S}$, define a function $\xi_4 = F_S(\xi_1, \xi_2, \xi_3)$, so that the line $\lambda(\xi_1, \xi_2, \xi_3, \xi_4)$ is tangent to S from below (F_S is only partially defined, and we put $F_S = +\infty$ when it is undefined; it is easily checked that F_S is algebraic of bounded degree and that its boundary is also algebraic of bounded degree). Let Φ be the lower envelope of the functions F_S, for $S \in \mathcal{S}$. Then a query line $\lambda(\xi_1, \xi_2, \xi_3, \xi_4)$ having the above properties misses all spheres of $\mathcal{S}$ if and only if $\xi_4 < \Phi(\xi_1, \xi_2, \xi_3)$. Thus, by Theorem 3.3,

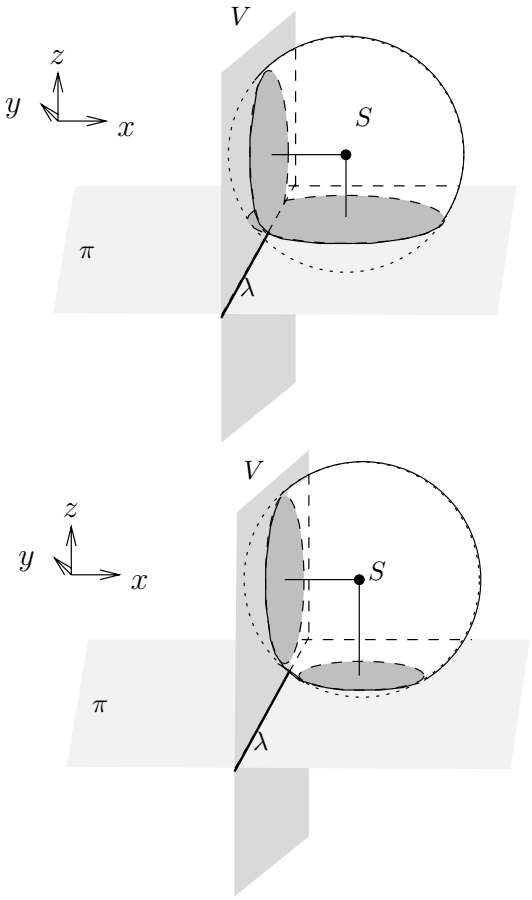

FIG. 1. *In the reduced subproblem, λ either intersects S or passes below it.*

one can answer such queries in time $O(\log^2 n)$, using $O(n^{3+\varepsilon})$ preprocessing time and storage, for any $\varepsilon > 0$. Plugging the half-space range-searching data structure and the point-location data structure into the multilevel data structure described in [AGPS, MS], we obtain a data structure for the segment-emptiness problem. A closer analysis of this data structure shows that the preprocessing time and storage of the overall data structure is still $O(n^{3+\varepsilon})$, for any $\varepsilon > 0$, and that the query time is $O(\log^2 n)$. Finally, plugging this data structure for segment-emptiness queries into the general ray-shooting technique of Agarwal and Matoušek [AMa], we obtain a final data structure, still requiring near-cubic storage and preprocessing, using which one can answer a ray-shooting query in time $O(\log^4 n)$. That is, we have shown the following.

THEOREM 4.1. *A set $\mathcal{S}$ of n spheres in $\mathbb{R}^3$ can be preprocessed in randomized expected time $O(n^{3+\varepsilon})$ into a data structure of size $O(n^{3+\varepsilon})$, for any $\varepsilon > 0$, so that a ray-shooting query can be answered in time $O(\log^4 n)$.*

4.2. Nearest-neighbor queries. Let $\mathcal{S} = \{s_1, \ldots, s_n\}$ be a collection of n objects ("sites") in 3-space, each having *constant description complexity*, namely, each defined by a constant number of algebraic equalities and inequalities of constant max-

imum degree. We wish to compute the *Voronoi diagram* $Vor(\mathcal{S})$ of $\mathcal{S}$ and preprocess it for efficient point-location queries. That is, each query specifies a point w in 3-space and seeks the site of $\mathcal{S}$ nearest to w (say, under the Euclidean distance). This is a generalization to 3-space of the classical *post-office problem*.

As observed in [ES], the problem is equivalent to the computation of the following lower envelope in 4-space. For each $s \in \mathcal{S}$, define $F_s(x, y, z)$ to be the Euclidean distance from (x, y, z) to s, and let Φ be the lower envelope of these functions. Then, given a query point (x, y, z), the site(s) $s \in \mathcal{S}$ nearest to w are those for which F_s attains Φ at (x, y, z). Thus, by Theorem 3.3, S can be preprocessed in $O(n^{3+\varepsilon})$ time, for any $\varepsilon > 0$, into a data structure of size $O(n^{3+\varepsilon})$, so that each query can be answered in $O(\log^2 n)$ time. (Note that Theorem 3.3 is indeed applicable here, because the functions F_s are all (piecewise) algebraic of constant maximum degree, as is easy to verify from the conditions assumed above.) Hence, we can conclude the following.

THEOREM 4.2. *A set $\mathcal{S}$ of n objects in $\mathbb{R}^3$, as described above, can be preprocessed in randomized expected time $O(n^{3+\varepsilon})$ into a data structure of size $O(n^{3+\varepsilon})$, for any $\varepsilon > 0$, so that a nearest-neighbor query can be answered in time $O(\log^2 n)$.*

This is a fairly general framework and admits numerous generalizations, e.g., we may replace the Euclidean distance by other distances, perform queries with objects other than points (as long as the location of a query object can be specified by only three real parameters; this is the case, e.g., if the query objects are translates of some rigid convex object), etc. Of course, any such generalization requires that the metric or the query objects also have constant description complexity, in the above sense. An interesting generalization is to *dynamic* nearest-neighbor queries in the plane, where each object of $\mathcal{S}$ moves along some given trajectory and each query (x, y, t) asks for the object of $\mathcal{S}$ nearest to the point (x, y) at time t. Using the same approach as above, this can be done, under appropriate assumptions on the shape and motion of the objects of $\mathcal{S}$, with $O(n^{3+\varepsilon})$ preprocessing time and storage, for any $\varepsilon > 0$ and $O(\log^2 n)$ query time.

Remarks. (i) In Theorem 4.2 we do not assume that the objects are pairwise disjoint. In this case one cannot hope to do much better, because the complexity of the Voronoi diagram of a set of intersecting objects in $\mathbb{R}^3$ is easily seen to be cubic in the worst case. For example, consider the Voronoi diagram of a set of planes in $\mathbb{R}^3$.

(ii) There is a prevailing conjecture that the complexity of generalized Voronoi diagrams of a set of *pairwise disjoint convex* objects in $\mathbb{R}^3$, and of dynamic Voronoi diagrams in the plane, is only near-quadratic, under reasonable assumptions concerning the distance function and the shape of the sites (and of their motions in the dynamic sense). This was indeed proved recently in [CKSW] for the case where the sites are lines in $\mathbb{R}^3$ and the distance function is induced by a convex polyhedron, and in [BSTY] for point sets under the L_1 or L_∞ metric. If this conjecture is established, then the above algorithm, of near-cubic cost, is far from being optimal for pairwise disjoint objects and will need to be improved considerably.

5. Applications: Batched problems. We next consider *batched* applications. These applications solve a variety of rather unrelated problems, but they all involve an almost identical subproblem, in which lower (or upper) envelopes in 4-space play a role. To avoid repetition, we describe in detail only one application and then state the improved bounds for the other applications without proof, referring the reader to the relevant literature.

The following applications are based on the *parametric-search* technique of Megiddo

[Me], which, in our case, requires a parallel algorithm for computing the lower envelope of a set of surfaces. However, the algorithm presented above is highly unparallelizable, because it uses an incremental insertion procedure of regions into 2-dimensional arrangements and later uses a plane-sweep in 3-space. To finesse this difficulty, we apply the parametric-search technique with an additional twist that avoids the need for a parallel algorithm.

5.1. Width in 3-space. Let S be a set of n points in 3-space. The *width* of S is the smallest distance between a pair of parallel planes so that the closed slab between the planes contains S. In two dimensions, the problem can be easily solved in $O(n \log n)$ time. An $O(n^2)$-time algorithm for three dimensions was presented in [HT]. Recently, Chazelle et al. [CEGSb] presented an $O(n^{8/5+\varepsilon})$-time algorithm for any $\varepsilon > 0$. In this subsection, we present an improved randomized algorithm, whose expected running time is $O(n^{17/11+\varepsilon}) = O(n^{1.546})$.

Clearly, it suffices to compute the width of the convex hull of S, so assume that the points of S are in convex position and that the convex hull $\mathcal{P}$ of S is given. It is known that any two planes defining the width of S are such that either one touches a face of $\mathcal{P}$ and one touches a vertex of $\mathcal{P}$, or each of them touches an edge of $\mathcal{P}$; see [CEGSb, HT]. The first case can be handled in $O(n \log n)$ time [CEGSb, HT]. The difficult case is to compute the smallest distance between a pair of parallel planes supporting $\mathcal{P}$ at two "antipodal" edges because, in the worst case, there can be $\Theta(n^2)$ such pairs of edges. Chazelle et al. [CEGSb] presented an $O(n^{8/5+\varepsilon})$-time algorithm to find the smallest distance, using the parametric-search technique. We will present a randomized algorithm, whose expected running time is $O(n^{17/11+\varepsilon})$, for any $\varepsilon > 0$, using a somewhat different approach.

Let $\mathcal{M}$ denote the *Gaussian diagram* (or *normal diagram*) of $\mathcal{P}$. $\mathcal{M}$ is a spherical map on the unit sphere $\mathbb{S}^2 \subset \mathbb{R}^3$. The vertices of $\mathcal{M}$ are points on $\mathbb{S}^2$, each being the outward normal of a face of $\mathcal{P}$, the edges of $\mathcal{M}$ are great circular arcs, each being the locus of the outward normal directions of all planes supporting $\mathcal{P}$ at some fixed edge, and the faces of $\mathcal{M}$ are regions, each being the locus of the outward normal directions of all planes supporting $\mathcal{P}$ at a vertex. $\mathcal{M}$ can be computed in linear time from $\mathcal{P}$. Let $\mathcal{M}'$ denote the spherical map $\mathcal{M}$ reflected through the origin, and consider the superposition of $\mathcal{M}$ and $\mathcal{M}'$. It suffices to consider the top parts of $\mathcal{M}$ and $\mathcal{M}'$, i.e., their portions within the hemisphere $z \geq 0$. Each intersection point between an edge of $\mathcal{M}$ and an edge of $\mathcal{M}'$ gives us a direction $\mathbf{u}$ for which there exist two parallel planes orthogonal to $\mathbf{u}$ and supporting $\mathcal{P}$ at two so-called "antipodal" edges. Thus the problem reduces to that of finding an intersection point for which the distance between the corresponding parallel supporting planes is minimized.

We centrally project the edges of (the top portions of) $\mathcal{M}$ and $\mathcal{M}'$ onto the plane $z = 1$. Each edge projects to a line segment or a ray. Let $\mathcal{E}$ and $\mathcal{E}'$ be the resulting sets of segments and rays in the plane. Using the algorithm described in [CEGSa], we can decompose $\mathcal{E} \times \mathcal{E}'$, in time $O(n \log^2 n)$, into a family of "canonical subsets"

$$\mathcal{F} = \{(\mathcal{E}_1, \mathcal{E}'_1), (\mathcal{E}_2, \mathcal{E}'_2), \ldots, (\mathcal{E}_t, \mathcal{E}'_t)\}\,, \tag{1}$$

such that

(i) $\mathcal{E}_i \subseteq \mathcal{E}$ and $\mathcal{E}'_i \subseteq \mathcal{E}'$;
(ii) $\sum_{i=1}^{t} (|\mathcal{E}_i| + |\mathcal{E}'_i|) = O(n \log^2 n)$;
(iii) each segment in $\mathcal{E}_i$ intersects every segment of $\mathcal{E}'_i$; and
(iv) for every pair (e, e') of intersecting segments in $\mathcal{E} \times \mathcal{E}'$, there is an i such that $e \in \mathcal{E}_i$ and $e' \in \mathcal{E}'_i$.

By property (iv), it suffices to consider each pair $(\mathcal{E}_i, \mathcal{E}'_i)$ separately. Let $\mathcal{L}_i$ (resp., $\mathcal{L}'_i$) denote the set of lines containing the edges of $\mathcal{P}$ corresponding to the edges of $\mathcal{E}_i$ (resp., $\mathcal{E}'_i$). By construction, all lines of $\mathcal{L}_i$ lie above all lines of $\mathcal{L}'_i$. Property (iii) implies that every pair of parallel planes π, π', where π contains a line λ of $\mathcal{L}_i$ and π' contains a line λ' of $\mathcal{L}'_i$, are supporting planes of $\mathcal{P}$. Hence, we want to compute the smallest distance between two such planes. Since the distance between π and π', as above, is equal to the distance between λ and λ', it follows that this problem is equivalent to that of computing the closest pair of lines in $\mathcal{L}_i \times \mathcal{L}'_i$. Hence, we need to solve the following problem: Given a set $\mathcal{L}$ of m "red" lines and another set $\mathcal{L}'$ of n "blue" lines in $\mathbb{R}^3$, such that all red lines lie above all blue lines, compute the closest pair of lines in $\mathcal{L} \times \mathcal{L}'$. For any pair of lines $\ell, \ell' \in \mathbb{R}^3$, let $d(\ell, \ell')$ be the Euclidean distance between them, and let

$$d(\mathcal{L}, \mathcal{L}') = \min_{\ell \in \mathcal{L},\, \ell' \in \mathcal{L}'} d(\ell, \ell')\,.$$

We first describe a simple randomized divide-and-conquer algorithm for computing $\delta^* = d(\mathcal{L}, \mathcal{L}')$, whose expected running time is $O(n^{3+\varepsilon} + m \log^2 n)$. If $m = O(1)$, we compute $d(\ell, \ell')$ for all pairs $\ell \in \mathcal{L}$, $\ell' \in \mathcal{L}'$ and choose the minimum distance. Otherwise, the algorithm performs the following steps:

(i) Choose a random subset $R_1 \subseteq \mathcal{L}$ of $m/2$ red lines; each subset of size $m/2$ is chosen with equal probability.
(ii) Solve the problem recursively for $(R_1, \mathcal{L}')$. Let $\delta_1 = d(R_1, \mathcal{L}')$.
(iii) Compute the set $R_2 = \{\ell \in \mathcal{L} \setminus R_1 \mid d(\ell, \mathcal{L}') < \delta_1\}$.
(iv) Compute $d(\ell, \ell')$ for all pairs $\ell \in R_2$, $\ell' \in \mathcal{L}'$, and output the minimum distance (or output δ_1 if R_2 is empty).

For a line $\ell' \in \mathcal{L}'$, let $R^{(\ell')} = \{\ell \in \mathcal{L} \mid d(\ell, \ell') < \delta_1\}$, so that $R_2 = \bigcup_{\ell' \in \mathcal{L}'} R^{(\ell')}$. Using a standard probabilistic argument, we can show that the expected size of $R^{(\ell')}$ is $O(1)$, for each $\ell' \in \mathcal{L}'$. Therefore the expected size of R_2 is $O(n)$. Consequently, the expected running time of step (iv) is $O(n^2)$. The only nontrivial step in the above algorithm is step (iii). We compute R_2 as follows. We map each line $\ell \in \mathcal{L}$ to a point $\psi(\ell) = (a_1, a_2, a_3, a_4)$ in $\mathbb{R}^4$, where $y = a_1 x + a_2$ is the equation of the xy-projection of ℓ, and $z = a_3 u + a_4$ is the equation of ℓ in the vertical plane $y = a_1 x + a_2$ (here u denotes the axis orthogonal to the z-axis). We can also map a line ℓ' to a surface $\gamma = \gamma(\ell')$ in this parameter space, which is the locus of all points $\psi(\ell)$ such that $d(\ell, \ell') = \delta_1$ and ℓ lies above ℓ'. Our choice of parameters ensures that $\gamma(\ell')$ is $x_1x_2x_3$-monotone; i.e., any line parallel to the x_4-axis intersects $\gamma(\ell')$ in at most one point. For any point lying below $\gamma(\ell')$ the corresponding line ℓ either lies below ℓ' or lies above ℓ' and $d(\ell, \ell') < \delta_1$. For a point lying above $\gamma(\ell')$, the corresponding line ℓ lies above ℓ' and $d(\ell, \ell') > \delta_1$. In view of the above discussion, $\gamma(\ell')$ is the graph of a partial function $x_4 = f_{\ell'}(x_1, x_2, x_3)$. (The function $f_{\ell'}$ is undefined only at points (x_1, x_2, x_3) that represent lines whose xy-projection is parallel to that of ℓ'; the locus of these points is a plane.) Let

$$F(x_1, x_2, x_3) = \max_{\ell' \in \mathcal{L}'} f_{\ell'}(x_1, x_2, x_3)$$

be the upper envelope of the set $\{f_{\ell'} \mid \ell' \in \mathcal{L}'\}$. For a line $\ell \in \mathcal{L}$ with $\psi(\ell) = (a_1, a_2, a_3, a_4)$, we have $a_4 \geq F(a_1, a_2, a_3)$ if and only if $d(\{\ell\}, \mathcal{L}') \geq \delta_1$. The problem of computing R_2 thus reduces to that of computing the set of points $\psi(\ell)$, for $\ell \in \mathcal{L} \setminus R_1$, that lie below the upper envelope F. By Theorem 3.3, this can be accomplished in

time $O(n^{3+\varepsilon} + m\log^2 n)$, for any $\varepsilon > 0$. The total time spent in steps (iii) and (iv) is thus $O(n^{3+\varepsilon} + m\log^2 n)$. Let $T(m,n)$ denote the maximum expected running time of the algorithm. Then

$$T(m,n) = \begin{cases} O(n) & \text{if } m \le m_0\,, \\ T\left(\frac{m}{2}, n\right) + O(n^{3+\varepsilon} + m\log^2 n) & \text{if } m > m_0\,, \end{cases}$$

where m_0 is a constant. The solution to this recurrence is easily seen to be

$$T(m,n) = O(n^{3+\varepsilon}\log m + m\log^2 n).$$

Hence, we can conclude the following.

LEMMA 5.1. *Given a set $\mathcal{L}$ of m lines and another set $\mathcal{L}'$ of n lines in $\mathbb{R}^3$, such that all lines in $\mathcal{L}$ lie above all lines of $\mathcal{L}'$, the closest pair of lines in $\mathcal{L}\times\mathcal{L}'$ can be computed in time $O(n^{3+\varepsilon}\log m + m\log^2 n)$ for any $\varepsilon > 0$.*

Next, we describe another algorithm for computing $\delta^* = d(\mathcal{L},\mathcal{L}')$, which uses the above procedure as a subroutine. We first describe an algorithm for the "fixed-size" problem that, given a parameter δ, determines whether $\delta^* < \delta$, $\delta^* = \delta$, or $\delta^* > \delta$. Next, we apply the parametric-search technique to this algorithm, with an additional twist, to compute $d(\mathcal{L},\mathcal{L}')$.

Consider the fixed-size problem. If $m > n^{3+\varepsilon}$, the problem can be solved in $O(m\log^2 n)$ time by computing δ^* using Lemma 5.1 and then comparing δ^* with δ. So assume that $m \le n^{3+\varepsilon}$.

We now map each line $\ell' \in \mathcal{L}'$ to the point $\psi(\ell')$, as defined above, and each line $\ell \in \mathcal{L}$ to a surface $\phi(\ell)$, which is the locus of all points $\psi(\ell')$ such that $d(\ell,\ell') = \delta$ and ℓ' lies below ℓ. (This is a dual representation of the problem, where the roles of $\mathcal{L}$ and of $\mathcal{L}'$ are interchanged.) The open region ϕ^- lying below $\phi(\ell)$ consists of points corresponding to lines ℓ' that lie below ℓ and satisfy $d(\ell,\ell') > \delta$. Again, each surface $\phi(\ell)$ is $x_1x_2x_3$-monotone; i.e., $\phi(\ell)$ is the graph of a partial function $x_4 = g_\ell(x_1,x_2,x_3)$. Let

$$G(x_1,x_2,x_3) = \min_{\ell\in\mathcal{L}} g_\ell(x_1,x_2,x_3)$$

be the lower envelope of these functions. By the same argument as above, $d(\mathcal{L},\mathcal{L}') < \delta$ if and only if there is a point $\psi(\ell')$, for some $\ell' \in \mathcal{L}'$, that lies above the lower envelope G, and $d(\mathcal{L},\mathcal{L}') = \delta$ if and only if no point corresponding to the lines of $\mathcal{L}'$ lies above G and at least one such point lies on G. To detect these two conditions, we proceed as follows.

(i) Fix a sufficiently large constant r. Choose a random subset $R \subset \mathcal{L}$ of size $t = cr\log r$, where c is some appropriate constant independent of r.

(ii) Let $\Gamma_R = \{g_\ell \mid \ell \in R\}$ be the set of t trivariate functions corresponding to the lines of R, and let G_R be the lower envelope of Γ_R. Decompose the region of $\mathbb{R}^4$ lying below G_R into a collection $\Xi = \{\tau_1, \ldots, \tau_k\}$ of $k = O(t^4\beta(t)) = O(r^4\beta(r)\log^4 r)$ semialgebraic cells of constant description complexity each; here $\beta(r)$ is a slowly growing function whose exact form is determined by the algebraic degree of the surfaces in Γ. Such a decomposition can be obtained using the algorithms described in [AST, CEGSb]. It is based on a decomposition of $\mathbb{R}^3$ into a family Ξ' of $O(r^4\beta(r)\log^4 r)$ cells of constant description complexity, so that, within each cell, the same set of

function graphs appear on the lower envelope G_R. The decomposition Ξ is then defined as the following collection of semiunbounded "prisms":

$$\Xi = \Big\{ \{(\mathbf{x}, z) \mid \mathbf{x} \in \Delta, z \le G_R(\mathbf{x})\} \mid \Delta \in \Xi' \Big\}.$$

(iii) For each prism $\tau \in \Xi$, let $\mathcal{L}_\tau = \{\ell \in \mathcal{L} \mid \phi(\ell) \cap \tau \neq \emptyset\}$ and $\mathcal{L}'_\tau = \{\ell' \in \mathcal{L}' \mid \psi(\ell') \in \tau\}$.

(iv) If $\bigcup_{\tau \in \Xi} \mathcal{L}'_\tau \neq \mathcal{L}'$ (i.e., there is a line $\ell' \in \mathcal{L}'$ such that the point $\psi(\ell')$ lies above G_R), then return $\delta > \delta^*$. If there is a $\tau \in \Xi$ such that $|\mathcal{L}_\tau| > m/r + 1$, then go back to Step (i).

(v) For each prism $\tau \in \Xi$, if both $\mathcal{L}_\tau, \mathcal{L}'_\tau$ are nonempty, then solve the problem recursively for $\mathcal{L}_\tau$ and $\mathcal{L}'_\tau$. If there is a subproblem for which $\delta > d(\mathcal{L}_\tau, \mathcal{L}'_\tau)$, then return $\delta > \delta^*$. If there is no such subproblem, but there is one subproblem for which $\delta = \delta^*$, then return $\delta = \delta^*$. Finally, if none of these two cases occur, then return $\delta < \delta^*$.

The correctness of the algorithm follows from the above observations. As for the running time, the well-known results on random sampling [ClS, HW] imply that if c is chosen sufficiently large then, with high probability, $|\mathcal{L}_\tau| \le m/r + 1$ for every $\tau \in \Xi$. Hence, the expected number of times we have to go back to step (i) from step (iv) is only a constant. Let $T(m, n)$ denote the expected running time of the algorithm. Then

$$T(m,n) = \begin{cases} O(m \log^2 n) & \text{if } m \ge n^{3+\varepsilon}, \\ \displaystyle\sum_{i=1}^{k} T\Big(\frac{m}{r} + 1, n_i\Big) + O(m+n) & \text{if } m < n^{3+\varepsilon}, \end{cases}$$

where $\sum_i n_i = n$ and $k = O(r^4 \beta(r) \log^4 r)$. The solution of the above recurrence is easily seen to be

$$T(m,n) = O(m^{8/11+\varepsilon'} n^{9/11+\varepsilon'} + m^{1+\varepsilon'} + n^{1+\varepsilon'})$$

for any $\varepsilon' > \varepsilon$.

Next, we describe how to apply the parametric-search technique to this algorithm. As mentioned earlier, we cannot use this algorithm directly in the parametric-search paradigm, because we do not know how to parallelize the first algorithm. We therefore simulate the above sequential algorithm at $\delta = \delta^*$ with the additional twist that, instead of just simulating the solution of the fixed-size problem at δ^*, the algorithm will attempt to compute δ^* explicitly. Since r is chosen to be constant, the set Ξ can be computed by making only $r^{O(1)}$ implicit "sign tests" involving δ^*, thus only a constant number, $r^{O(1)}$, of solutions of the fixed-size problem are needed; see [AST, CEGSb] for details. It can be checked that the problem of determining whether a surface $\phi(\ell)$ intersects a prism $\tau \in \Xi$ or whether a point $\psi(\ell')$ lies in τ can be reduced to computing the signs of a constant number of univariate polynomials, each of constant degree, at δ^* (this follows since τ has constant description complexity). Let $\Pi = \{p_1(\delta), \ldots, p_s(\delta)\}$, where $s = O(m+n)$, be the set of these polynomials, over all lines of $\mathcal{L}, \mathcal{L}'$ and over all prisms of Ξ. Let $\delta_1 < \cdots < \delta_u$ be the real roots of these polynomials, where u is also $O(m+n)$. By a binary search over these roots we compute the largest root δ_α such that $\delta_\alpha \le d(\mathcal{L}, \mathcal{L}')$. Each step of the binary search involves comparing δ^* with some δ_i, which in turn involves solving an instance of the fixed-size problem for some δ_i. If $\delta_\alpha = \delta^*$, we have found the value of δ^* so we stop right away. Thus,

assume that $\delta_\alpha < \delta^*$. We can now easily resolve the signs of all polynomials of Π by evaluating them at $(\delta_\alpha + \delta_{\alpha+1})/2$. Once we have computed $\mathcal{L}_\tau, \mathcal{L}'_\tau$ for all $\tau \in \Xi$, we compute $d(\mathcal{L}_\tau, \mathcal{L}'_\tau)$ recursively and return $\min_{\tau\in\Xi} d(\mathcal{L}_\tau, \mathcal{L}'_\tau)$ as $d(\mathcal{L}, \mathcal{L}')$. (Note that $\bigcup_{\tau\in\Xi} \mathcal{L}'_\tau = \mathcal{L}'$, since we are simulating the algorithm at δ^*.) The correctness of the algorithm is established by the following lemma.

LEMMA 5.2. *If none of the calls to the fixed-size problem in steps* (i)–(iv) *returns the value of* $d(\mathcal{L}, \mathcal{L}')$, *then* $d(\mathcal{L}, \mathcal{L}') = \min_{\tau\in\Xi} d(\mathcal{L}_\tau, \mathcal{L}'_\tau)$.

Proof. Let $\delta^* = d(\mathcal{L}, \mathcal{L}')$, and let $\ell \in \mathcal{L}$, $\ell' \in \mathcal{L}'$ be a pair of lines such that $d(\ell, \ell') = \delta^*$. Since we are simulating the fixed-size problem at $\delta = \delta^*$, no point corresponding to the lines of $\mathcal{L}'$ lies above the lower envelope of the surfaces defined by $\mathcal{L}$ (for $\delta = \delta^*$), that is, there exists a prism $\tau \in \Xi$ such that $\ell' \in \mathcal{L}'_\tau$ (i.e., $\psi(\ell')$ lies in τ). If ℓ does not belong to $\mathcal{L}_\tau$, then $\psi(\ell')$ lies below $\phi(\ell)$, which implies that $d(\ell, \ell') > \delta^*$, a contradiction. Hence the pair (ℓ, ℓ') appears in the subproblem involving $\mathcal{L}_\tau$ and $\mathcal{L}'_\tau$. □

Let $T'(m, n)$ denote the maximum expected running time of the algorithm. Then we obtain the following recurrence

$$T'(m,n) = \begin{cases} O(m \log^2 n) & \text{if } m \geq n^{3+\varepsilon}, \\ \displaystyle\sum_{i=1}^{O(r^4\beta(r)\log^4 r)} T'\Big(\frac{m}{r} + 1, n_i\Big) + r^{O(1)} O(m^{8/11+\varepsilon} n^{9/11+\varepsilon} + m^{1+\varepsilon} + n^{1+\varepsilon}) & \\ & \text{if } m < n^{3+\varepsilon}. \end{cases}$$

The solution of the above recurrence is also

$$T'(m,n) = O(m^{8/11+\varepsilon'} n^{9/11+\varepsilon'} + m^{1+\varepsilon'} + n^{1+\varepsilon'})$$

for a different but still arbitrarily small constant $\varepsilon' > \varepsilon$.

Hence, we obtain the following result.

LEMMA 5.3. *Given a set* $\mathcal{L}$ *of* m *lines and another set* $\mathcal{L}'$ *of* n *lines such that all lines in* $\mathcal{L}$ *lie above all lines of* $\mathcal{L}'$, *the closest pair of lines in* $\mathcal{L} \times \mathcal{L}'$ *can be computed in randomized expected time* $O(m^{8/11+\varepsilon} n^{9/11+\varepsilon} + m^{1+\varepsilon} + n^{1+\varepsilon})$ *for any* $\varepsilon > 0$.

Finally, we apply Lemma 5.3 to all subsets $(\mathcal{E}_i, \mathcal{E}'_i)$ of $\mathcal{F}$ (see (1)) and output the minimum of the distances obtained for each subproblem. This is equal to the minimum distance between any pair of parallel planes, each supporting an edge of $\mathcal{P}$. The total expected running time is

$$\sum_{i=1}^{t} O\left(|\mathcal{E}_i|^{8/11+\varepsilon} |\mathcal{E}'_i|^{9/11+\varepsilon} + |\mathcal{E}_i|^{1+\varepsilon} + |\mathcal{E}'_i|^{1+\varepsilon}\right) = O(n^{17/11+\varepsilon'}),$$

where ε' is yet another but still arbitrarily small positive constant.

Putting everything together, we can conclude the following.

THEOREM 5.4. *The width of a set of* n *points in* $\mathbb{R}^3$ *can be computed in randomized expected time* $O(n^{17/11+\varepsilon})$, *for any* $\varepsilon > 0$.

Remark. Informally speaking, the "ugly" exponent 17/11 is the result of an interaction between the exponent 3, appearing in the bound of Theorem 3.3, and the exponent 4, appearing in the bound for the size of the vertical decomposition Ξ used above. After the original submission of this paper, Agarwal and Sharir [ASb] observed that, for the width problem and for the two problems studied below, one can show that the size of this vertical decomposition is only near cubic in the number of

surfaces. This in turn leads to an improved analysis of the above algorithm and implies that its running time is only $O(n^{3/2+\varepsilon})$ for any $\varepsilon > 0$ (and a similar improvement applies to the two subsequent algorithms given below). Agarwal and Sharir [ASb] gave a different randomized algorithm for these problems, without using parametric searching.

5.2. Biggest stick in a simple polygon. Let P be a simple polygon in the plane having n edges. We wish to find the longest line segment e that is contained in (the closed set) P. Chazelle and Sharir presented in [CS] an $O(n^{1.99})$-time algorithm for this problem, which was later improved by Agarwal, Sharir, and Toledo [AST] to $O(n^{8/5+\varepsilon})$ for any $\varepsilon > 0$. The latter paper reduces this problem, just as in the computation of the width of a point set in 3-space, to that of finding the extreme value of a certain function of two parameters, optimized over all intersection points between the edges of two straight-edge planar maps. This problem, in turn, can be reduced to the problem of optimizing the function over all intersection points of pairs of lines in $\mathcal{L}_1 \times \mathcal{L}_2$, where $\mathcal{L}_1, \mathcal{L}_2$ are two appropriate families of lines in the plane. By regarding the lines of $\mathcal{L}_1$ as data points, and each line of $\mathcal{L}_2$ as inducing a certain function over the lines of $\mathcal{L}_1$, and by using an appropriate parametrization of these points and functions, the problem can be reduced to that of testing whether any point in a certain set of points in 4-space lies above the lower envelope of a certain collection of trivariate functions. Combining the approach of [AST] with the one described in the previous subsection, we can compute the longest segment e in expected time $O(n^{17/11+\varepsilon})$ for any $\varepsilon > 0$. Omitting all the details, which can be found in [AST], we conclude the following.

THEOREM 5.5. *One can compute the biggest stick that fits inside a simple polygon with n edges, in randomized expected time $O(n^{17/11+\varepsilon})$, for any $\varepsilon > 0$.*

5.3. Minimum-width annulus. Let S be a set of n points in the plane. We wish to find the (closed) annulus of smallest width that contains S, i.e., we want to compute two concentric disks D_1 and D_2 of radii r_1 and r_2, such that all points of S lie in the closure of $D_2 \setminus D_1$ and $r_2 - r_1$ is minimized. This problem has been studied in [AST]. It is known [AST, EFNN] that the center of such an annulus is a vertex of the closest-point Voronoi diagram, $\mathrm{Vor}_c(S)$, of S, a vertex of the farthest-point Voronoi diagram, $\mathrm{Vor}_f(S)$, of S, or the intersection point of an edge of $\mathrm{Vor}_c(S)$ and an edge of $\mathrm{Vor}_f(S)$. The difficult part is testing the intersection points of edges of the two diagrams, because there can be $\Theta(n^2)$ such points in the worst case. Following the same idea as in [AST], which reduces the problem to that of batched searching of points relative to an envelope of functions in four dimensions, but using the technique described above for computing the width in 3-space, we obtain the following result (see [AST] for details).

THEOREM 5.6. *The smallest-width annulus containing a set of n points in the plane can be computed in randomized expected time $O(n^{17/11+\varepsilon})$, for any $\varepsilon > 0$.*

REFERENCES

[AMa] P. AGARWAL AND J. MATOUŠEK, *Ray shooting and parametric search,* SIAM J. Comput., 22 (1993), pp. 794–806.

[AMb] P. AGARWAL AND J. MATOUŠEK, *Range searching with semialgebraic sets,* Discrete Comput. Geom., 11 (1994), pp. 393–418.

[ASS] P. AGARWAL, O. SCHWARZKOPF, AND M. SHARIR, *The overlay of lower envelopes and its applications,* Discrete Comput. Geom., 15 (1996), pp. 1–13.

[ASa] P. Agarwal and M. Sharir, *On the number of views of polyhedral terrains,* Discrete Comput. Geom., 12 (1994), pp. 177–182.

[ASb] P. Agarwal and M. Sharir, *Efficient randomized algorithms for some geometric optimization problems,* Discrete Comput. Geom., 16 (1996), pp. 317–337.

[AST] P. Agarwal, M. Sharir, and S. Toledo, *New applications of parametric searching in computational geometry,* J. Algorithms, 17 (1994), pp. 292–318.

[AGPS] P. Agarwal, L. Guibas, M. Pellegrini, and M. Sharir, *Ray Shooting among Spheres,* manuscript, 1992.

[ArS] B. Aronov and M. Sharir, *Triangles in space, or: Building (and analyzing) castles in the air,* Combinatorica, 10 (1990), pp. 137–173.

[BDS+] J. D. Boissonnat, O. Devillers, R. Schott, M. Teillaud, and M. Yvinec, *Applications of random sampling to on-line algorithms in computational geometry,* Discrete Comput. Geom., 8 (1992), pp. 51–71.

[BD] J. D. Boissonnat and K. Dobrindt, *On-line randomized construction of the upper envelope of triangles and surface patches in* $\mathbb{R}^3$*,* Comput. Geom. Theory Appl., 5 (1996), pp. 303–320.

[BSTY] J. D. Boissonnat, M. Sharir, B. Tagansky, and M. Yvinec, *Voronoi diagrams in higher dimensions under certain polyhedral convex distance functions,* in Proc. 11th ACM Symp. Comput. Geom., Vancouver, 1995, ACM, New York, pp. 79–88.

[CEGSa] B. Chazelle, H. Edelsbrunner, L. Guibas, and M. Sharir, *Algorithms for bichromatic line segment problems and polyhedral terrains*, Algorithmica, 11 (1994), pp. 116–132.

[CEGSb] B. Chazelle, H. Edelsbrunner, L. Guibas, and M. Sharir, *Diameter, width, closest line pair, and parametric searching,* Discrete Comput. Geom., 10 (1993), pp. 183–196.

[CEGSS] B. Chazelle, H. Edelsbrunner, L. Guibas, M. Sharir, and J. Snoeyink, *Computing a single face in an arrangement of line segments and related problems,* SIAM J. Computing, 22 (1993), pp. 1286–1302.

[CS] B. Chazelle and M. Sharir, *An algorithm for generalized point location and its applications,* J. Symbolic Comput., 10 (1990), pp. 281–309.

[CKSW] L. P. Chew, K. Kedem, M. Sharir, B. Tagansky, and E. Welzl, *Voronoi diagrams of lines in three dimensions under a polyhedral convex distance function,* in Proc. 6th ACM-SIAM Symp. on Discrete Algorithms, San Francisco, 1995, SIAM, Philadelphia, pp. 197–204.

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

[dBDS] M. de Berg, K. Dobrindt, and O. Schwarzkopf, *On lazy randomized incremental construction,* Discrete Comput. Geom., 14 (1995), pp. 261–286.

[EFNN] H. Ebara, N. Fukuyama, H. Nakano, and Y. Nakanishi, *Roundness algorithms using the Voronoi diagrams,* in First Canadian Conf. Comput. Geom., abstract, 1989.

[ES] H. Edelsbrunner and R. Seidel, *Voronoi diagrams and arrangements,* Discrete Comput. Geom., 1 (1986), pp. 25–44.

[HS] D. Halperin and M. Sharir, *New bounds for lower envelopes in* 3 *dimensions, with applications to visibility in terrains,* Discrete Comput. Geom., 12 (1994), pp. 313–326.

[HW] D. Haussler and E. Welzl, ϵ*-nets and simplex range queries*, Discrete Comput. Geom., 2 (1987), pp. 127–151.

[HRR] J. Heintz, T. Recio, and M.-F. Roy, *Algorithms in real algebraic geometry and applications to computational geometry,* in Discrete and Computational Geometry: Papers from the DIMACS Special Year, J.E. Goodman, R. Pollack, and W. Steiger, eds., AMS Press, Providence, RI, 1991, pp. 137–163.

[HT] M. Houle and G. Toussaint, *Computing the width of a set,* IEEE Trans. Pattern Matching and Machine Intelligence, 5 (1988), pp. 761–765.

[Me] N. Megiddo, *Applying parallel computation algorithms in the design of serial algorithms,* J. Assoc. Comput. Mach., 30 (1983), pp. 852–865.

[MS] S. Mohaban and M. Sharir, *Ray shooting amidst spheres in three dimensions,* SIAM J. Comput., 26 (1997), pp. 654–674.

[Mua] K. Mulmuley, *A fast planar partition algorithm*, I, J. Symbolic Comput., 10 (1990), pp. 253–280.

[Mub] K. Mulmuley, *A fast planar partition algorithm*, II, J. Assoc. Comput. Mach., 38 (1991), pp. 74–103.

[PS] J. Pach and M. Sharir, *The upper envelope of piecewise linear functions and the bound-*

ary of a region enclosed by convex plates: Combinatorial analysis, Discrete Comput. Geom., 4 (1989), pp. 291–309.

[PT] F. P. Preparata and R. Tamassia, *Efficient point location in a convex spatial cell-complex,* SIAM J. Comput., 21 (1992), pp. 267–280.

[Sha] M. Sharir, *Almost tight upper bounds for lower envelopes in higher dimensions,* Discrete Comput. Geom., 12 (1994), pp. 327–345.

[SA] M. Sharir and P. Agarwal, *Davenport-Schinzel Sequences and Their Geometric Applications*, Cambridge University Press, New York, 1995.