Dynamic Planar Voronoi Diagrams for General Distance Functions and their Algorithmic Applications*

Haim Kaplan† Wolfgang Mulzer‡ Liam Roditty§ Paul Seiferth¶ Micha Sharir‖

Abstract

We describe a new data structure for dynamic nearest neighbor queries in the plane with respect to a general family of distance functions that includes L_p-norms and additively weighted Euclidean distances, and for general (convex, pairwise disjoint) sites that have constant description complexity (line segments, disks, etc.). Our data structure has a polylogarithmic update and query time, improving an earlier data structure of Agarwal, Efrat and Sharir that required $O(n^\varepsilon)$ time for an update and $O(\log n)$ time for a query [1]. Our data structure has numerous applications, and in all of them it gives faster algorithms, typically reducing an $O(n^\varepsilon)$ factor in the bounds to polylogarithmic. To further demonstrate its effectiveness, we give here two new applications: an efficient construction of a spanner in a disk intersection graph, and a data structure for efficient connectivity queries in a dynamic disk graph.

To obtain this data structure, we combine and extend various techniques and obtain several side results that are of independent interest. Our data structure depends on the existence and an efficient construction of *"vertical" shallow cuttings* in arrangements of bivariate algebraic functions. We prove that an appropriate level in an arrangement of a random sample of a suitable size provides such a cutting. To compute it efficiently, we develop a randomized incremental construction algorithm for finding the lowest k levels in an arrangement of bivariate algebraic functions (we mostly consider here collections of functions whose lower envelope has linear complexity, as is the case in the dynamic nearest-neighbor context). To analyze this algorithm, we improve a longstanding bound on the combinatorial complexity of the vertical decomposition of these levels. Finally, to obtain our structure, we plug our vertical shallow cutting construction into Chan's algorithm for efficiently maintaining the lower envelope of a dynamic set of planes in $\mathbb{R}^3$. While doing this, we also revisit Chan's technique and present a variant that uses a single binary counter, with a simpler analysis and an improved amortized deletion time.

1 Introduction

Nearest neighbor searching in the plane is one of the most fundamental problems in computational geometry, and it has been studied since the very beginning of the field [5,22]. Given a set S of *sites* in the plane, we would like to construct a data structure that allows us to find the "closest" site for any given query object. If the set of sites is fixed, Voronoi diagrams and their many variants provide a simple and well-understood solution to this problem [3, 5]. However, in many applications, the set of sites may change dynamically, as we insert and delete sites into/from S, and we want to answer nearest-neighbor queries interleaved with the updates. This setting is much less understood.

If S consists of points and the distances are measured in the Euclidean metric, it is known how to achieve polylogarihmic update and query time [9]. However, there are many more geometric problems in which dynamic nearest neighbor searching forms a key component in an efficient solution, and in which it is crucial to perform nearest neighbor queries for more general distance functions (e.g., L_p-norms or additively weighted Euclidean distances). These applications include the problems of dynamically maintaining a bichromatic closest pair of sites, a minimum-weight Euclidean red-blue matching, a Euclidean minimum spanning tree, the intersection of unit balls in three dimensions, or the smallest stabbing disk of a family of simply shaped compact strictly-convex sets in the plane. Another recent application is an algorithm for computing shortest path trees in unit-disk graphs (see Section 7 for more details and references). Despite the wide range of applications, there has been virtually no progress on dynamic nearest neighbor search in the plane for general distance func-

*A full version is available on the arXiv [20]. Work by Haim Kaplan, Wolfgang Mulzer, Liam Roditty, and Paul Seiferth has been supported by grant 1161/2011 from the German-Israeli Science Foundation. Work by Haim Kaplan has also been supported by grant 1841-14 from the Israel Science Foundation, and by the Israeli Centers for Research Excellence (I-CORE) program (center no. 4/11). Work by Wolfgang Mulzer and Paul Seiferth has also been supported by grant MU/3501/1 from Deutsche Forschungsgemeinschaft (DFG). Work by Micha Sharir has been supported by Grant 2012/229 from the U.S.-Israel Binational Science Foundation, by Grant 892/13 from the Israel Science Foundation, by the Israeli Centers for Research Excellence (I-CORE) program (center no. 4/11), and by the Hermann Minkowski–MINERVA Center for Geometry at Tel Aviv University.

†Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel; haimk@tau.ac.il.

‡Institut für Informatik, Freie Universität Berlin, Berlin 14195, Germany; mulzer@inf.fu-berlin.de.

§Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel; liamr@macs.biu.ac.il.

¶Institut für Informatik, Freie Universität Berlin, Berlin 14195, Germany; pseiferth@inf.fu-berlin.de.

‖Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel; michas@tau.ac.il.

tions since the last millennium. The state of the art solution dates from 1999 and provides $O(n^{\varepsilon})$ update time with $O(\log n)$ time queries [1]. We describe a new data structure that improves this bound to polylogarithmic update and query time for a wide range of distance functions. For this, we need to bring together a diverse set of techniques such as randomized incremental construction, relative (p, ε)-approximations, shallow cuttings for xy-monotone surfaces in $\mathbb{R}^3$, and data structuring tricks.

We now give a more detailed description of the setting. Let $S \subset \mathbb{R}^2$ be a set of n pairwise disjoint sites, each being a simply-shaped compact convex region in the plane (such as points, line segments, disks, etc.), and let δ be some given continuous distance function between points in the plane. For a site $s \in S$, define a function $f_s : \mathbb{R}^2 \to \mathbb{R}$ by $f_s(x, y) = \delta((x, y), s)$, namely, $f_s(x, y) = \min\{\delta((x, y), p) \mid p \in s\}$ (compactness of the objects in S, and continuity of δ, ensure that the minimum exists). We assume that δ and the sites in S have constant description complexity, i.e., they are defined by a constant number of polynomial equations and inequalities of constant maximum degree. Let F denote the collection of the bivariate functions $\{f_s\}_{s \in S}$. The *lower envelope* $\mathcal{E}_F$ of F is the pointwise minimum $\mathcal{E}_F(x, y) = \min_{s \in S} f_s(x, y)$, and its xy-projection is called the *minimization diagram* of F, and is denoted as $\mathcal{M}_F$. The *combinatorial complexity* of $\mathcal{E}_F$ or of $\mathcal{M}_F$ is the number of their vertices, edges and faces. See [24] for a detailed treatment of these concepts.

Finding the δ-nearest neighbor in S of a query point $q \in \mathbb{R}^2$ calls for identifying the site s for which $\mathcal{E}_F(q) = f_s(q)$. Such a query translates to a *vertical ray shooting* query in $\mathcal{E}_F$, where we seek the intersection point of the z-vertical line through q with $\mathcal{E}_F$, or, alternatively, we want to locate q in $\mathcal{M}_F$, where each face φ of this planar map is labeled with the site s for which f_s attains the minimum over φ.

The structure and complexity of $\mathcal{E}_F$ and of $\mathcal{M}_F$, as well as algorithms for their construction and manipulation, have been studied extensively for several decades (again, see [24]). Briefly, under the assumptions made above, the combinatorial complexity of $\mathcal{E}_F$, measured in terms of the number of vertices, edges, and faces of this surface (or of its corresponding minimization diagram) is $O(n^{2+\varepsilon})$, for any $\varepsilon > 0$ (where the constant of proportionality depends on ε). But in many interesting special cases, the most ubiquitous of which is the case where the functions f_s are all linear (i.e., their graphs are non-vertical planes), the complexity of $\mathcal{E}_F$ is only linear in n. The case of planes arises, after some trivial algebraic manipulations, for point sites under the Euclidean distance. Then, $\mathcal{M}_F$ is the Euclidean Voronoi diagram of S. There are many variants of Voronoi diagrams for other classes of sites and other distance functions, for which the complexity of $\mathcal{E}_F$ remains linear; see, e.g., the recent book by Aurenhammer, Klein, and Lee [3].

Assuming linear complexity of $\mathcal{E}_F$, and the availability of an efficient algorithm for constructing it, all we need to do, in the so-called "static" case, is to preprocess $\mathcal{M}_F$ for fast planar point location, and then locate each query point in $\mathcal{M}_F$, in $O(\log n)$ time.

However, when the sites in S can be inserted or deleted, this corresponds to the setup in which F changes dynamically, by insertions and deletions of functions. The main issue in this situation is that, upon an insertion or a deletion of a function, $\mathcal{E}_F$ might change rather drastically, and maintaining an explicit representation of $\mathcal{M}_F$ after each update might be overwhelmingly expensive. The goal, pursued in this paper, as well as in several earlier works (reviewed below) is to store some implicit representation of $\mathcal{E}_F$ that still supports efficient execution of vertical ray shooting queries for the current envelope (or point location queries in the current $\mathcal{M}_F$).

In all applications of dynamic nearest neighbor searching mentioned above, the lower envelope of the corresponding set F of bivariate functions has linear complexity. The distance functions that arise are typically L_p-metrics, for some $1 \leq p \leq \infty$, or additively weighted Euclidean metrics, where each (say, point) site $s \in S$ has an associated weight $w_s \in \mathbb{R}$, and $\delta(q, s) = |qs| + w_s$, where $|qs|$ is the Euclidean distance between q and s. See, e.g., [3, 21] for details concerning the linear complexity of the envelope in these cases. This property of having a lower envelope of linear complexity also holds for general classes of pairwise-disjoint compact convex sites of *constant description complexity*.

Our main result is an efficient data structure that dynamically maintains a set F of bivariate functions, of the kind described above, under insertions and deletions of functions, and supports efficient vertical ray shooting queries into the lower envelope of F. Assuming, as above, that the complexity of the lower envelope is linear, the worst-case cost of a query, as well as the amortized cost of an update, is polylogarithmic in our data structure. Applying this data structure, we obtain faster solutions to all the applications mentioned above, and more, essentially replacing an $O(n^{\varepsilon})$ factor in the complexity of earlier solutions by a polylogarithmic factor.

A brief context. Consider first the case where the graphs of the bivariate functions in F are planes. (As already noted, this case corresponds to the dynamic nearest neighbor problem for a set S of point sites with respect to the Euclidean metric.) A classic solution for this special case is due to Agarwal and Matoušek [2].

They show how to maintain dynamically, in an implicit manner, the lower envelope of a set F of at most n planes, with amortized update time $O(n^\varepsilon)$, where $\varepsilon > 0$ can be made arbitrarily small (and where the constant of proportionality depends on ε); vertical ray shooting queries take $O(\log n)$ worst-case time. The case of more general bivariate functions, of the sort considered in this paper, was studied by Agarwal et al. [1]. In cases where the complexity of the lower envelope is linear (such as those reviewed above), the technique of Agarwal et al. [1] has amortized update (insertion or deletion) time $O(n^\varepsilon)$, and worst-case query time $O(\log n)$, matching the known bounds for planes by Agarwal and Matoušek [2].

For more than ten years after the work of Agarwal and Matoušek [2], it was open whether the $O(n^\varepsilon)$ update time can be improved. In SODA 2006, Chan [9] presented an ingenious construction, in which both the (amortized) update time and the (worst-case) query time are polylogarithmic, for the case of planes. More precisely, Chan's data structure (combined with the recent deterministic construction of shallow cuttings by Chan and Tsakalidis [11]) supports insertions in $O(\log^3 n)$ amortized time, deletions in $O(\log^6 n)$ amortized time, and queries in $O(\log^2 n)$ worst-case time. However, up to now it was not known whether a similar result (with polylogarithmic update time) is possible for arbitrary bivariate functions of constant description complexity with linear envelope complexity. In this paper we settle this question, by providing an algorithm that meets all these performance goals. Along the way, we also improve the deletion time for Chan's data structure for the case of planes by a logarithmic factor and the bound of Agarwal et al. [1] for the complexity of the vertical decomposition of the $(\leq k)$ level in an arrangement of surfaces in $\mathbb{R}^3$ by a factor of k^ε.

2 Overview of Our Techniques

As already mentioned in the introduction, our dynamic nearest neighbor structure for general distance functions requires a whole zoo of techniques that need to be tamed and brought together carefully. We now first give a broad overview of how these techniques interact, and then we provide a more detailed description on how we use them. See Figure 1 for an illustration; the various concepts that appear in the figure will be explained in the subsequent text.

Maybe the most crucial observation is that all the geometry required in Chan's data structure for dynamic lower envelopes of planes in $\mathbb{R}^3$ lies in the construction of small-sized *shallow cuttings* for planes (of a certain special type). Thus, once we have small-sized shallow cuttings for surfaces, we are able to maintain dynamically the lower envelope of surfaces, or equivalently, solve the generalized dynamic nearest neighbor problem in the plane. It turns out that using relative (p, ε)-approximations, we can find the required cutting quite easily. However, at this point we do not know how to obtain the conflict lists for such a cutting in an efficient manner. To solve this issue, we give an algorithm that is based on randomized incremental construction (RIC) to compute the $(\leq k)$-level in an arrangement of surfaces. This algorithm can be used to efficiently compute the required shallow cutting and its conflict lists. Together with an improved version of Chan's dynamic lower envelope structure, this gives the generalized nearest neighbor data structure. We show the impact of this structure by providing several applications (with new bounds), old and new. In what follows, we describe the main ideas of the specific parts in more detail. The full version with all proofs and algorithms can be found on the arXiv [20].

3 Preliminaries

Let $\mathcal{F}$ be a family of bivariate functions in $\mathbb{R}^3$, and let F be a finite subset of $\mathcal{F}$. We assume that the functions in $\mathcal{F}$ are continuous, totally defined, and algebraic, and that they have *constant description complexity*, formally meaning that the graph of each function is a semialgebraic set, defined by a constant number of polynomial equalities and inequalities of constant maximum degree. The *lower envelope* $\mathcal{E}_F$ of F is the graph of the pointwise minimum of the functions of F. The xy-projection of $\mathcal{E}_F$ is a subdivision of the xy-plane called the *minimization diagram* $\mathcal{M}_F$ of F. Each of its faces corresponds to (and is labeled by) the function in F that attains $\mathcal{E}_F$ over that face.

When $\mathcal{M}_F$ consists of $O(|F|)$ faces, vertices, and edges, for any finite $F \subseteq \mathcal{F}$, we say that $\mathcal{F}$ has lower envelopes of *linear complexity*. In particular, this holds when $\mathcal{F}$ is the family of all nonvertical planes, and when $\mathcal{F}$ is a family of distance functions under some metric (or so-called convex distance function), each of which measures the distance of a point in the xy-plane to some given site. Throughout, we will only consider families $\mathcal{F}$ of linear complexity.

For simplicity, we also assume that F is in *general position*, i.e., no more than three function graphs meet at a common point, no more than two function graphs meet in a one-dimensional curve, and no pair of graphs are tangent to each other. (This holds if, say, the coefficients of the polynomials defining the functions in F are algebraically independent over the reals [24].) Furthermore, we assume that the coordinate frame is generic, so that the xy-projections of the intersection curves of pairs of the function graphs are also in general

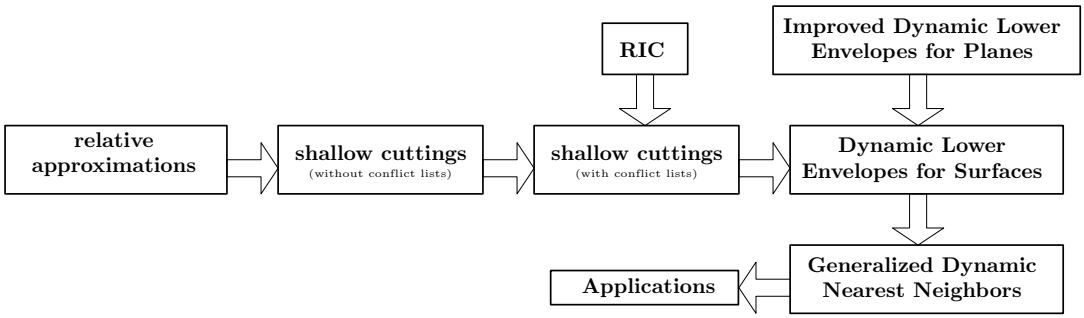

Figure 1: An overview of how our techniques relate to each other.

position, defined in an analogous sense.

4 Shallow Cuttings for Surfaces

As mentioned above, the geometric core of Chan's data structure lies in an efficient construction of small-sized *shallow cuttings* of a particularly favorable kind, called *vertical* shallow cuttings [8, 11]. To define these constructs, let $\mathcal{A}(F)$ be the arrangement of a set of n bivariate functions $F \subseteq \mathcal{F}$ in $\mathbb{R}^3$. The *level* of a point $q \in \mathbb{R}^3$ in $\mathcal{A}(F)$ is the number of functions of F whose graphs pass strictly below q. For $k \in \{0, \ldots, n-1\}$, the *k-level* $L_k(F)$ of $\mathcal{A}(F)$ is the closure of the set of points at level k that lie on the union of the graphs of the functions in F. We denote by $L_{\leq k}(F)$ the union of the first k levels of $\mathcal{A}(F)$. For given parameters k, $r < n$, a *k-shallow $(1/r)$-cutting* in $\mathcal{A}(F)$ is a collection Λ of pairwise openly disjoint regions τ, each of constant description complexity, so that the union of these regions covers $L_{\leq k}(F)$, and so that the interior of each region $\tau \in \Lambda$ is intersected by at most n/r graphs of functions of F. The *size* of Λ is the number of regions in Λ.

A *vertical* k-shallow $(1/r)$-cutting in $\mathcal{A}(F)$ is a collection Λ of pairwise openly disjoint vertical semi-unbounded *pseudo-prisms*, a notion to be defined momentarily, so that, as above, the union of these pseudo-prisms covers $L_{\leq k}(F)$, and so that the interior of each pseudo-prism $\tau \in \Lambda$ is intersected by at most n/r graphs of functions of F. Note that, for both conditions to hold simultaneously, we must have $k \leq n/r$. In our setting, we will always have $r = \Theta(n/k)$, which is the case most relevant for applications.

A pseudo-prism τ consists of all points that lie vertically below some portion $\overline{\tau}$ of a graph of a function in F, so that $\overline{\tau}$ has constant description complexity. In our application, $\overline{\tau}$ will be a *pseudo-trapezoid*, defined as the portion of a function graph consisting of points (x, y, z) satisfying $x^- \leq x \leq x^+$, $\psi^-(x) \leq y \leq \psi^+(x)$, for real numbers $x^- < x^+$ and for (semi-)algebraic functions ψ^-, ψ^+ of constant description complexity. In the case of planes, $\overline{\tau}$ will simply be a triangle, and we do not insist that $\overline{\tau}$ be contained in one of the input planes.

Chan [8] was the first to show the existence of vertical shallow cuttings for planes in three dimensions. Such a cutting is associated with a polyhedral triangulated xy-monotone terrain, which lies entirely above the k-level of the arrangement, so that each triangle $\overline{\tau}$ of the terrain generates a semi-unbounded triangular prism with $\overline{\tau}$ as its top face. Such shallow cuttings have many applications, in particular in Chan's dynamic lower envelope data structure [9]. The deterministic construction of Chan and Tsakalidis [11] constructs vertical shallow cuttings. Recently, Har-Peled, Kaplan, and Sharir [17] gave an alternative construction with some additional favorable properties.[1]

Essentially, once a fast construction of vertical shallow cuttings of sufficiently small size is available, we can plug it into the machinery developed by Chan [9], in an almost black-box fashion, to obtain a fast data structure for dynamic maintenance of $\mathcal{E}_F$ in the general setting. Agarwal et al. [1] prove the existence of shallow cuttings of optimal size for general functions, but their cuttings are not "vertical", in the above sense, and a direct algorithmic implementation of their construction yields an additional $O(n^\varepsilon)$ factor for both the size and the construction time of the cutting. When applied to the dynamic maintenance problem, this makes the (amortized) cost of an update $O(n^\varepsilon)$ rather than polylogarithmic. Refining this bound is one of the main goals of the present paper.

To achieve this, we design a different algorithm for computing a vertical shallow cutting, obtaining several technical results that we consider of independent interest. We first use the notion of *relative approximation*,

[1] One significant difference is that the "top terrain" in [17] approximates the corresponding level k up to any specified accuracy, whereas the structure in [11] does not.

as in Har-Peled and Sharir [18], to conclude that, by choosing a random sample S_k of size $\frac{cn}{\varepsilon^2 k}$ from F, and by constructing the level t of $\mathcal{A}(S_k)$, where t is in the range $\left[(1+\frac{\varepsilon}{3})\lambda,\ (1+\frac{\varepsilon}{2})\lambda\right]$, for $\lambda = \frac{c\log n}{\varepsilon^2}$ and c being a suitable absolute constant, we get an *ε-approximation* of level k of $\mathcal{A}(F)$, with high probability. This means that any such level t of $\mathcal{A}(S_k)$ lies between levels k and $(1+\varepsilon)k$ of $\mathcal{A}(F)$. Furthermore, we show that if we choose t uniformly at random in the above range, the expected complexity of the corresponding level is $O\left(\frac{n}{\varepsilon^5 k}\log^2 n\right)$.

Having computed such a level, we project it onto the xy-plane, compute the standard planar vertical decomposition of the projection, lift each trapezoid φ of this decomposition back to a trapezoidal-like subface φ^* on the original level, and associate with it the semi-unbounded vertical prism consisting of all points that lie vertically below φ^*, see Figure 2. We denote this set of prisms by Λ_k. The following lemma shows that Λ_k is a vertical k-shallow (k/n)-cutting. All proofs can be found in the full version.

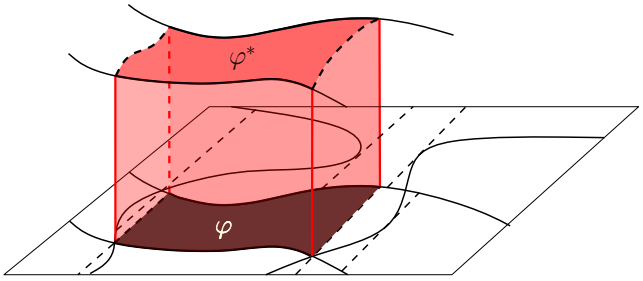

Figure 2: A pseudo-trapezoid φ (dark) lifted up to its original position φ^* in $\mathbb{R}^3$ (lighter) yields a semi-unbounded vertical pseudo-prism.

LEMMA 4.1. *Λ_k is a shallow cutting of the first k levels of $\mathcal{A}(F)$. It consists (in expectation) of*

$$|\Lambda_k| = O\left(\frac{n}{\varepsilon^5 k}\log^2 n\right)$$

prisms. Each prism in Λ_k intersects at least k and at most $(1+2\varepsilon)k$ graphs of functions of F.

Our next hurdle is to efficiently compute Λ_k, together with the *conflict lists* of its prisms, where, for a prism $\tau \in \Lambda_k$, the conflict list $\mathrm{CL}(\tau)$ of τ is the set of all functions $f \in F$ that cross the interior of τ. (Note that, although the construction of Λ_k is performed with respect to the sample S_k, the conflict lists of its prisms are defined with respect to the entire collection F.)

5 Randomized Incremental Construction

Let $t \in \{0, \ldots, n-1\}$ be a parameter. We now consider the classic problem of computing the $t+1$ shallowest levels in the arrangement $\mathcal{A}(F)$. A standard approach to this construction is via *randomized incremental construction* (RIC, in short); see, e.g., [5, 22]. For this, one adds the functions in F one by one, in a random order, and maintains a representation of the desired structure (the first $t+1$ levels in our case) of the subset of functions inserted so far. Let $F_i \subseteq F$ be the functions we have inserted after i steps. After each insertion we maintain the vertical decomposition of $L_{\le t} = L_{\le t}(F_i)$ of the first $t+1$ levels of $\mathcal{A}(F_i)$. This vertical decomposition is denoted by $\mathrm{VD}_{\le t}$ and is defined in the following standard manner.

We obtain $\mathrm{VD}_{\le t}$ in two decomposition stages. In the first stage, we erect within each cell C of $L_{\le t}$ a vertical curtain up and/or down from each edge (an intersection edge of a pair of surfaces) of $L_{\le t}$. Each such curtain consists of maximal vertical segments contained in (the closure of) C and passing through the points of the edge. The collection of these curtains partitions C into subcells. Each subcell has at most one *ceiling* ("top" facet), at most one *floor* ("bottom" facet), and all other facets lie on its vertical curtains; the ceiling or the floor may be missing if the subcell is unbounded.

The complexity of a subcell may still be arbitrarily large. Thus, in the second stage, we take each subcell C', project it onto the xy-plane, and apply to the projection a similar but two-dimensional vertical decomposition: we erect a y-vertical segment from each vertex and from each locally x-extreme point on the edges. This yields a collection of pseudo-trapezoidal subcells. We then lift each such pseudo-trapezoid vertically to 3-space; formally, we take each trapezoid τ and form the intersection $(\tau \times \mathbb{R}) \cap C'$. This yields a decomposition of C' into *pseudo-prisms*, each with "constant description complexity", as defined above. The desired vertical decomposition consists of all pseudo-prisms obtained in this manner for all of $L_{\le t}$. More details can be found in [12, 24].

To analyze our RIC, we need to bound the complexity of $\mathrm{VD}_{\le t}$. The following crucial lemma improves an earlier bound of $O(nt^{2+\varepsilon})$ by Agarwal et al. [1]. The parameter s in the lemma is defined as follows. For any quadruple f, g, f', g' of functions of F, we let $s(f, g, f', g')$ denote the number of co-vertical pairs of points $q \in f \cap g$, $q' \in f' \cap g'$. We define s to be the maximum value of $s(f, g, f', g')$, over all such quadruples. By our assumptions on $\mathcal{F}$ (including general position), we have $s = O(1)$. The function $\lambda_{s+2}(t)$ in the lemma is the familiar bound on the maximum length of a Davenport-Schinzel sequence of order $s+2$ [24].

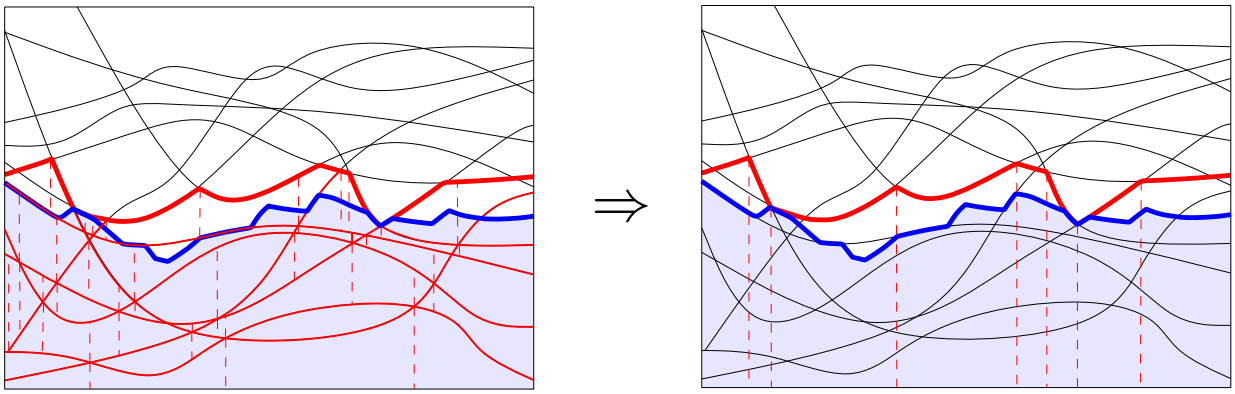

Figure 3: The $(\leq t)$-level of $\mathcal{A}(S_k)$ (red) is decomposed by $\mathrm{VD}_{\leq t}(S_k)$ (left). Furthermore, $\mathrm{VD}_{\leq t}(S_k)$ covers the $(\leq k)$-level of $\mathcal{A}(F)$ (blue), and we can combine the cells in $\mathrm{VD}_{\leq t}(S_k)$ to obtain a vertical k-shallow (k/n)-cutting for $\mathcal{A}(F)$ together with its conflict lists (right).

LEMMA 5.1. *Let F be a set of n functions from $\mathcal{F}$, and let $1 \leq t \leq n-1$ be a parameter. The complexity of $\mathrm{VD}_{\leq t}(F)$ is $O(nt\lambda_{s+2}(t))$.*

Lemma 5.1 allows us to analyze our RIC for the $t+1$ shallowest levels in $\mathcal{A}(F)$. This yields the following theorem.

THEOREM 5.1. *Let $1 \leq t \leq n-1$. The first $t+1$ levels of an arrangement of the graphs of n continuous totally defined algebraic functions of constant description complexity, for which the complexity of the lower envelope of any m functions is $O(m)$, can be constructed by a randomized incremental algorithm, whose expected running time is $O(nt\lambda_{s+2}(t)\log(n/t)\log n)$, and whose expected storage is $O(nt\lambda_{s+2}(t))$.*

Using our randomized incremental algorithm, we construct a vertical shallow cutting of the first $k+1$ levels in $\mathcal{A}(F)$, consisting of $O\left(\frac{n}{k}\log^2 n\right)$ prisms, each with a conflict list of size $O(k)$. The approach is as follows: we run the RIC with parameter $t \approx \frac{c\log n}{\varepsilon^2}$. We stop after inserting the first $\frac{cn}{\varepsilon^2 k}\log n$ functions. This can be regarded as drawing the desired sample $S_k \subseteq F$ as in Section 4. By Lemma 4.1 the t-level of S_k yields a vertical shallow cutting. In addition, we obtain a vertical decomposition $\mathrm{VD}_{\leq t}(S_k)$ of the $t+1$ shallowest levels of $\mathcal{A}(S_k)$ and the conflict lists (with respect to the whole set F) of its cells. Each prism in $\mathrm{VD}_{\leq t}(S_k)$ extends between two consecutive levels of the present arrangement $\mathcal{A}(S_k)$, so the prisms cannot be used to form a vertical shallow cutting. Nevertheless, as we show in the full version, it is possible to transform our decomposition into a vertical shallow cutting, including the new conflict lists of its semi-unbounded prisms. This process is depicted for two-dimensional arrangements in Figure 3. This gives the following theorem.

THEOREM 5.2. *Let F be a set of n continuous totally defined algebraic functions of constant description complexity for which the complexity of the lower envelope of any m functions is $O(m)$. Furthermore, let $k \in \{1,\ldots,n\}$. Then, there is a vertical shallow cutting Λ_k for the first k levels of $\mathcal{A}(F)$ with the following properties:*

1. *The number of cells in Λ_k is $O((n/k)\log^2 n)$.*
2. *Each prism in Λ_k intersects at least k and at most $2k$ graphs of functions in F.*
3. *We can find Λ_k and the conflict lists for its prisms in expected $O(n\log^3 n\lambda_{s+2}(\log n))$ time using expected $O(n\log n\lambda_{s+2}(\log n))$ space.*

6 Improvement of Chan's Data Structure

Once we have available an efficient mechanism for constructing vertical shallow cuttings and their conflict lists as in Theorem 5.2, we apply it to obtain our dynamic nearest neighbor data structure, adapting the technique of Chan for the case of planes to our setting [9]. While doing so, we re-examine the work of Chan, and we present it in a way that (in our opinion) is easier to understand. Our exposition follows a more standard route: we begin with a static data structure, then extend it to support insertions, using a variant of the well-known Bentley-Saxe binary counter technique [4], and finally show how to perform deletions via re-insertions of planes, using a so-called *deletion lookahead mechanism*, the major innovation in Chan's work. We believe that this new perspective sheds additional light on the inner workings of the structure. Along the way, we improve the amortized cost of a deletion, by a logarithmic factor, to $O(\log^5 n)$. This is achieved by a more aggressive

pruning strategy in the construction of the static data structure, combined with a new amortization argument that shows that not too many surfaces are pruned. Deletions are the costliest operations in Chan's technique, and are therefore the bottleneck in most applications. Our efforts are summarized in the following theorem.

THEOREM 6.1. *The lower envelope of a set of n bivariate functions of the type that we consider can be maintained dynamically, to support insertions, deletions, and queries, so that each insertion takes $O(\log^5 n\lambda_{s+2}(\log n))$ amortized expected time, each deletion takes $O(\log^9 n\lambda_{s+2}(\log n))$ amortized expected time, and each query takes $O(\log^2 n)$ worst-case deterministic time, where n is the number of functions in the data structure at the time the operation is performed. The data structure requires $O(n\log^3 n)$ expected storage.*

Remark. For the special case of non-vertical planes, we get (worst-case, deterministic) query time $O(\log^2 n)$, (amortized, deterministic) insertion time $O(\log^3 n)$, and (amortized, deterministic) deletion time $O(\log^5 n)$. The data structure requires $O(n)$ storage.

7 Applications

We give several applications of Theorem 6.1. As discussed above, for a finite set $S \subset \mathbb{R}^2$ of pairwise disjoint sites, finding for a point $q \in \mathbb{R}^2$ its nearest neighbor in S under any norm or convex distance function δ [13] translates to ray shooting in the lower envelope of the graphs of the functions $f_s(x) = \delta(x, s)$, $s \in S$. Thus, if these functions have lower envelopes of linear complexity, Theorem 6.1 yields a dynamic nearest neighbor data structure for S. Recall that the minimization diagram of the lower envelope of F is the Voronoi diagram of S under δ [3, 14]. Two classes of distance functions are of particular interest. First, let $p \in [1, \infty]$. We define for $(x_1, y_1), (x_2, y_2) \in \mathbb{R}^2$ the L_p *metric* $\delta_p((x_1, y_1), (x_2, y_2)) = (|x_1 - x_2|^p + |y_1 - y_2|^p)^{1/p}$. It is well known that δ_p is a metric, and that it induces lower envelopes of linear complexity for any set of sites as above [21]. Second, let $S \subset \mathbb{R}^2$ be a set of point sites, where each $s \in S$ has an associated weight $w_s \in \mathbb{R}$. We define a distance function $\delta : \mathbb{R}^2 \times S \to \mathbb{R}$ by $\delta(p, s) = w_s + |ps|$, where $|\cdot|$ denotes the Euclidean distance. This distance function also induces lower envelopes of linear complexity [3].

7.1 Direct Applications of Dynamic Nearest Neighbor Search There are numerous results that can be improved immediately with our new tools. See the table below for the new bounds.

Dynamic Bichromatic Closest Pair. Let δ be a planar distance function, and let $R, B \subset \mathbb{R}^2$ be two sets of planar point sites. A *bichromatic closest pair* is a pair in $R \times B$ that minimizes δ. The goal is to maintain a bichromatic closest pair under insertions and deletions of points. We can improve Theorem 6.8 in Agarwal et al. [1] by combining Eppstein's method [15] with Theorem 6.1.

Minimum Euclidean Bichromatic Matching. Let R and B be two sets of n points in the plane. A *minimum Euclidean bichromatic matching* of (R, B) is a set of n line segments between R and B such that each point in $R \cup B$ is incident to exactly one segment and such that the total length of the segments is minimum. Agarwal et al. [1, Theorem 7.1] show how to find such a set using a dynamic bichromatic closest pair data structure for the additively weighted Euclidean metric, building on a trick by Vaidya [25]. Thus, our previous application gives an improved bound.

Dynamic Minimum Spanning Trees. Let S be a set of sites, and T the minimum spanning tree w.r.t. to an L_p norm, $p \geq 1$. We want to maintain T explicitly as S changes dynamically. Following Eppstein [15], our first application immediately gives a data structure for this.

Maintaining the Intersection of Unit Balls in Three Dimensions. Let B be a set of unit balls in $\mathbb{R}^3$. We want to maintain the intersection $B^\cap$ of the balls in B under insertions and deletions, while supporting the following queries: (a) for any $p \in \mathbb{R}^3$, determine if $p \in B^\cap$, and (b) after performing each update, determine whether $B^\cap \neq \emptyset$. Agarwal et al. [1] use dynamic lower envelopes to maintain $B^\cap$. Their algorithm performs a query via parametric search in a black box fashion. Thus, using Theorem 6.1, we can improve Theorem 8.1 in Agarwal et al. [1].

Maintaining the Smallest Stabbing Disk. Let $\mathcal{C}$ be a family of simply shaped compact strictly-convex planar sets. We wish to dynamically maintain a finite subset $C \subseteq \mathcal{C}$ together with a smallest disk that intersects all the sets of C (see Agarwal et al. [1, Section 9] for precise definitions). Again, we can use Theorem 6.1 in a black-box fashion to improve on the previous result of Agarwal et al..

7.2 Range Reporting and Approximate Range Counting We also obtain new range searching results. Let $F \subset \mathcal{F}$, $|F| = n$, and $q \in \mathbb{R}^3$ a query point. First, we describe a data structure to find all surfaces of F that pass vertically below q. If F consists of planes, Chan [7] showed how to obtain a data structure with $O(n \log n)$ space and expected query time $O(\log n + k)$, where k is the output size. The core of the approach is a follows: take a sequence of random samples $R_0 \subset R_1 \subset \cdots \subset R_{\log n} = F$, with $|R_i| = 2^i$. For each

Problem	Old Bound	New Bound
Dynamic bichromatic closest pair in general planar metric	n^{ε} update [1]	$\log^{10} n\lambda_{s+2}(\log n)$ insertion, $\log^{11} n\lambda_{s+2}(\log n)$ deletion
Minimum Euclidean planar bichromatic matching	$n^{2+\varepsilon}$ [1]	$n^2 \log^{11} n\lambda_{s+2}(\log n)$
Dynamic minimum spanning tree in general planar metric	n^{ε} update [1]	$\log^{13} n\lambda_{s+2}(\log n)$ update
Dynamic intersection of unit balls in $\mathbb{R}^3$	n^{ε} update [1] queries in $\log n$ and $\log^4 n$	$\log^5 n\lambda_{s+2}(\log n)$ insertion, $\log^9 n\lambda_{s+2}(\log n)$ deletion, queries in $\log^2 n$ and $\log^5 n$
Dynamic Smallest Stabbing Disk	n^{ε} update [1] queries in $\log^4 n$ time	$\log^5 n\lambda_{s+2}(\log n)$ insertion, $\log^9 n\lambda_{s+2}(\log n)$ deletion, queries in $\log^5 n$

R_i, compute a set of pairwise interior disjoint vertical prisms that cover the region below the lower envelope of R_i, together with their conflict lists with respect to F. To do this efficiently, one performs a standard RIC for the lower envelope of F and stops after every 2^i-th insertion for $1 \leq i \leq \log n$, see [7] for details. For surfaces, we can carry out the same approach using our RIC algorithm with $t = O(1)$. The total construction time is $O(n \log^2 n)$. Following Chan's [7] analysis, we get the next theorem.

THEOREM 7.1. *Let $F \subset \mathcal{F}$ be the graphs of n continuous totally defined algebraic functions of constant description complexity, so that the complexity of the lower envelope of any m functions is linear. We can preprocess F in expected time $O(n \log^2 n)$ into a data structure of size $O(n \log n)$ that can answer the following queries: given a point $q \in \mathbb{R}^3$, find all surfaces $f \in F$ that pass strictly below q. The expected query time is $O(\log n + k)$, where k is the output size.*

Our next data structure is for *approximate range counting*. Given a query point $q \in \mathbb{R}^3$, we want to determine the number of surfaces in F strictly below q up to a prespecified multiplicative error of $(1+\varepsilon)$, i.e., we want to approximate the level of q. The following is an immediate adaptation of a result by Har-Peled et al. [17].

THEOREM 7.2. *Let $F \subset \mathcal{F}$ be a set of n surfaces in $\mathbb{R}^3$ and let $\varepsilon > 0$ be a parameter. We can construct a data structure of size $O((n/\varepsilon^6) \log^2 n)$ in expected time $O((n/\varepsilon^6) \log^3 n\lambda_{s+2}(\log n))$ that can answer approximate level queries in $\mathcal{A}(F)$, with relative error ε, in time $O(\log n \log((1/\varepsilon) \log n))$.*

7.3 Problems on Disk Intersection Graphs Disk intersection graphs constitute a particularly fruitful application domain for our data structure: let $S \subset \mathbb{R}^2$ be a finite set of point sites, each with an associated weight $w_p > 0$, $p \in S$. The *disk intersection graph* for S, $D(S)$, has S as vertex set and an edge pq between two sites p and q in S if and only if $|pq| \leq w_p + w_q$, i.e., if the w_p-disk around p intersects the w_q-disk around q. If all weights are 1, $D(S)$ is the *unit disk graph* for S, UD(S). Disk intersection graphs are a popular model for geometrically defined graphs and enjoy an increasing interest in the research community, driven by applications in wireless sensor networks [6,10,16,23].

Shortest Path Trees in Unit Disk Graphs. Cabello and Jejčič [6] find a shortest path tree in UD(S), for any given site $r \in S$, in time $O(n^{1+\varepsilon})$ using the bichromatic closest pair structure of Agarwal et al. [1, Theorem 6.8]. Our structure immediately yields an improvement.

Dynamic Connectivity in Disk Graphs. We show how to maintain $D(S)$ under insertions and deletions while answering *reachability queries* efficiently: given $s, t \in S$, is there a path in $D(S)$ from s to t? For this, we combine previous ideas for unit disk graphs [19] with a dynamic additively weighted nearest neighbor structure. The update time of our data structure depends on the *radius ratio* Ψ of the largest and the smallest weight of the sites. The previous bound of Chan, Pătraşcu, and Roditty [10] is only slightly sublinear, but independent of Ψ.

BFS Trees in Disk Graphs. As observed by Roditty and Segal [23] in the context of unit disk graphs, a dynamic Euclidean nearest neighbor structure can be used for computing exact BFS-trees in disk graphs, for

Problem	Old Bound	New Bound
Shortest path tree in a unit disk graph	$n^{1+\varepsilon}$ [6]	$n \log^{11} n\lambda_{s+2}(\log n)$
Dynamic connectivity in disk intersection graphs	$n^{20/21}$ update $n^{1/7}$ query [10]	$\Psi^2 \log^9 n\lambda_{s+2}(\log n)$ update $\log n/\log\log n$ query
BFS tree in a disk intersection graph	$n^{1+\varepsilon}$ [23]	$n \log^9 n\lambda_{s+2}(\log n)$
$(1+\rho)$-spanner for a disk intersection graph	$n^{4/3+\varepsilon}\rho^{-4/3}\log^{2/3}\Psi$ [16]	$(n/\rho^2)\log^9 n\lambda_{s+2}(\log n)$

any given root vertex. The same technique works for disk graphs, once a dynamic additively weighted nearest neighbor structure is available.

Spanners for Disk Graphs. Finally, we give an algorithm for computing a $(1+\rho)$-spanner in a disk intersection graph, for $\rho > 0$. A $(1+\rho)$-*spanner* for $D(S)$ is a subgraph H of $D(S)$ such that the shortest path distances in H approximate the shortest path distances in $D(S)$ up to a factor of $(1+\rho)$. The previous construction by Fürer and Kasiviswanathan [16] has a running time that depends on the radius ratio Ψ, as defined above. Our new algorithm is independent of Ψ and achieves almost linear running time, improving the previous algorithm by a factor of at least $n^{1/3}$.

Acknowledgments. We thank Pankaj Agarwal for suggesting several applications for our data structure. We would also like to thank the anonymous referees for helpful comments.

References

[1] Pankaj K. Agarwal, Alon Efrat, and Micha Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. *SIAM Journal on Computing*, 29(3):912–953, 1999.

[2] Pankaj K. Agarwal and Jiří Matoušek. Dynamic halfspace range reporting and its applications. *Algorithmica*, 13(4):325–345, 1995.

[3] Franz Aurenhammer, Rolf Klein, and Der-Tsai Lee. *Voronoi Diagrams and Delaunay Triangulations*. World Scientific Publishing, 2013.

[4] Jon Louis Bentley and James B. Saxe. Decomposable searching problems. I. Static-to-dynamic transformation. *J. Algorithms*, 1(4):301–358, 1980.

[5] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark H. Overmars. *Computational Geometry: Algorithms and applications*. Springer-Verlag, third edition, 2008.

[6] Sergio Cabello and Miha Jejčič. Shortest paths in intersection graphs of unit disks. *Comput. Geom.*, 48(4):360–367, 2015.

[7] Timothy M. Chan. Random sampling, halfspace range reporting, and construction of $(\leq k)$-levels in three dimensions. *SIAM Journal on Computing*, 30(2):561–575, 2000.

[8] Timothy M. Chan. Low-dimensional linear programming with violations. *SIAM Journal on Computing*, 34(4):879–893, 2005.

[9] Timothy M. Chan. A dynamic data structure for 3-D convex hulls and 2-D nearest neighbor queries. *J. ACM*, 57(3):Art. 16, 15, 2010.

[10] Timothy M. Chan, Mihai Pătraşcu, and Liam Roditty. Dynamic connectivity: connecting to networks and geometry. *SIAM Journal on Computing*, 40(2):333–349, 2011.

[11] Timothy M. Chan and Konstantinos Tsakalidis. Optimal deterministic algorithms for 2-d and 3-d shallow cuttings. *Discrete Comput. Geom.*, 56(4):866–881, 2016.

[12] Bernard Chazelle, Herbert Edelsbrunner, Leonidas J. Guibas, and Micha Sharir. A singly-exponential stratification scheme for real semi-algebraic varieties and its applications. In *Proc. 31st Internat. Colloq. Automata Lang. Program. (ICALP)*, pages 179–193, 1989.

[13] L. Paul Chew and Robert L. (Scot) Drysdale III. Voronoi diagrams based on convex distance functions. In *Proc. 1st Annu. Sympos. Comput. Geom. (SoCG)*, pages 235–244, 1985.

[14] Herbert Edelsbrunner and Raimund Seidel. Voronoi diagrams and arrangements. *Discrete Comput. Geom.*, 1(1):25–44, 1986.

[15] David Eppstein. Dynamic Euclidean minimum spanning trees and extrema of binary functions. *Discrete Comput. Geom.*, 13:111–122, 1995.

[16] Martin Fürer and Shiva Prasad Kasiviswanathan. Spanners for geometric intersection graphs with applications. *J. Comput. Geom.*, 3(1):31–64, 2012.

[17] Sariel Har-Peled, Haim Kaplan, and Micha Sharir. Approximating the k-level in three-dimensional plane arrangements. In *Proc. 27th Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, 2016.

[18] Sariel Har-Peled and Micha Sharir. Relative (p,ϵ)-approximations in geometry. *Discrete Comput. Geom.*,

45(3):462–496, 2011.

[19] Haim Kaplan, Wolfgang Mulzer, Liam Roditty, and Paul Seiferth. Dynamic connectivity for unit disk graphs. In *Proc. 32nd European Workshop Comput. Geom. (EWCG)*, 2016.

[20] Haim Kaplan, Wolfgang Mulzer, Liam Roditty, Paul Seiferth, and Micha Sharir. Dynamic planar Voronoi diagrams for general distance functions and their algorithmic applications. `arxiv:1604.03654`, 2016.

[21] D. T. Lee. Two-dimensional Voronoĭ diagrams in the L_p-metric. *J. ACM*, 27(4):604–618, 1980.

[22] Ketan Mulmuley. *Computational Geometry: An Introduction Through Randomized Algorithms*. Prentice-Hall, 1994.

[23] Liam Roditty and Michael Segal. On bounded leg shortest paths problems. *Algorithmica*, 59(4):583–600, 2011.

[24] Micha Sharir and Pankaj K. Agarwal. *Davenport-Schinzel sequences and their geometric applications*. Cambridge University Press, 1995.

[25] Pravin M. Vaidya. Geometry helps in matching. *SIAM Journal on Computing*, 18(6):1201–1225, 1989.