Vertical Decomposition in 3D and 4D with Applications to Line Nearest-Neighbor Searching in 3D *

Pankaj K. Agarwal† Esther Ezra‡ Micha Sharir§

November 6, 2023

Abstract

Vertical decomposition is a widely used general technique for decomposing the cells of arrangements of semi-algebraic sets in $\mathbb{R}^d$ into constant-complexity subcells. In this paper, we settle in the affirmative a few long-standing open problems involving the vertical decomposition of substructures of arrangements for $d = 3, 4$: (i) Let $\mathcal{S}$ be a collection of n semi-algebraic sets of constant complexity in $\mathbb{R}^3$, and let $U(m)$ be an upper bound on the complexity of the union $\mathcal{U}(\mathcal{S}')$ of any subset $\mathcal{S}' \subseteq \mathcal{S}$ of size at most m. We prove that the complexity of the vertical decomposition of the complement of $\mathcal{U}(\mathcal{S})$ is $O^*(n^2 + U(n))$ (where the $O^*(\cdot)$ notation hides subpolynomial factors). We also show that the complexity of the vertical decomposition of the entire arrangement $\mathcal{A}(\mathcal{S})$ is $O^*(n^2 + X)$, where X is the number of vertices in $\mathcal{A}(\mathcal{S})$. (ii) Let $\mathcal{F}$ be a collection of n trivariate functions whose graphs are semi-algebraic sets of constant complexity. We show that the complexity of the vertical decomposition of the portion of the arrangement $\mathcal{A}(\mathcal{F})$ in $\mathbb{R}^4$ lying below the lower envelope of $\mathcal{F}$ is $O^*(n^3)$.

These results lead to efficient algorithms for a variety of problems involving these decompositions, including algorithms for constructing the decompositions themselves, and for constructing $(1/r)$-cuttings of substructures of arrangements of the kinds considered above. One additional algorithm of interest is for output-sensitive point enclosure queries amid semi-algebraic sets in three or four dimensions.

In addition, as a main domain of applications, we study various proximity problems involving points and lines in $\mathbb{R}^3$: We first present a linear-size data structure for answering nearest-neighbor queries, with points, amid n lines in $\mathbb{R}^3$ in $O^*(n^{2/3})$ time per query. We also study the converse problem, where we return the nearest neighbor of a query line amid n input points, or lines, in $\mathbb{R}^3$. We obtain a data structure of $O^*(n^4)$ size that answers a nearest-neighbor query in $O(\log n)$ time.

1 Introduction

Let $\mathcal{S}$ be a family of n semi-algebraic sets[1] of constant complexity in $\mathbb{R}^d$. The *arrangement* of $\mathcal{S}$, denoted by $\mathcal{A}(\mathcal{S})$, is the decomposition of $\mathbb{R}^d$ into maximal connected relatively open cells of all dimensions, so that all points within a cell lie in the relative interior or boundary of the same subfamily of sets of $\mathcal{S}$. Because of their wide range of applications, arrangements of semi-algebraic sets have been extensively studied [14, 20]. The combinatorial complexity of a cell in $\mathcal{A}(\mathcal{S})$ can be quite large, and its topology can be quite complex [14], so a fundamental problem in the area of arrangements, for both combinatorial and algorithmic applications, is to decompose a cell of $\mathcal{A}(\mathcal{S})$ into constant-complexity subcells, each homeomorphic to a ball. In some applications, we wish to decompose all cells of $\mathcal{A}(\mathcal{S})$ while in others only a subset of cells of $\mathcal{A}(\mathcal{S})$.

*Work by Pankaj Agarwal has been partially supported by NSF grants IIS-18-14493, CCF-20-07556, and CCF-22-23870. Work by Esther Ezra has been partially supported by Israel Science Foundation Grant 800/22, and also by US-Israel Binational Science Foundation under Grant 2022131. Work by Micha Sharir has been partially supported by Israel Science Foundation Grant 260/18.

†Department of Computer Science, Duke University, Durham, NC 27708, USA; pankaj@cs.duke.edu, https://orcid.org/0000-0002-9439-181X

‡School of Computer Science, Bar Ilan University, Ramat Gan, Israel; ezraest@cs.biu.ac.il, https://orcid.org/0000-0001-8133-1335

§School of Computer Science, Tel Aviv University, Tel Aviv, Israel; michas@tauex.tau.ac.il, http://orcid.org/0000-0002-2541-3763

[1]Roughly speaking, a semi-algebraic set in $\mathbb{R}^d$ is the set of points in $\mathbb{R}^d$ that satisfy a Boolean formula over a set of polynomial inequalities; the complexity of a semi-algebraic set is the number of polynomials defining the set and their maximum degree. See [20] for formal definitions of a semi-algebraic set and its dimension.

Vertical decomposition is a popular general technique (and perhaps the only general-purpose technique) for constructing such a decomposition. Roughly speaking, vertical decomposition recurses on the dimension d. Let C be a cell of $\mathcal{A}(\mathcal{S})$. For $d = 2$, the vertical decomposition of C is obtained by erecting a y-vertical segment up and down from each vertex of C and from each point of vertical tangency on the boundary of C, and extending these segments till they hit another edge of C, or else all the way to infinity. This results in a decomposition of C into vertical *pseudo-trapezoids* (trapezoids, for short). For $d = 3$, we first erect, upwards and downwards, z-vertical *curtains* from each edge of C and from the silhouette (the locus of points with z-vertical tangency) of each 2-face of C, and extend them until they hit ∂C (or else all the way to infinity). The resulting subcells have a unique pair of faces as their "floor" and "ceiling," but their complexity can still be large. In the second decomposition phase, we project each subcell onto the xy-plane, apply planar vertical decomposition to the projection, and lift each resulting subcell (trapezoid) vertically up to $\mathbb{R}^3$ to the range between the floor and ceiling of the original subcell. This results in a decomposition of C into vertical *pseudo-prisms* (prisms for short), each bounded by up to six facets. This recursive scheme (on the dimension) can be generalized to higher dimensions, but it becomes more involved as the dimension grows. In this work, though, we only use the three- and four-dimensional scenarios. See [24, 36, 45].

Vertical decompositions, similar to some other geometric decomposition schemes, provide a mechanism for constructing geometric *cuttings* of various substructures of arrangements of semi-algebraic sets [14], which in turn leads to an efficient divide-and-conquer mechanism for solving a variety of combinatorial and algorithmic problems, as well as for constructing data structures for geometric searching problems [10]. The performance of these algorithms and data structures depends on the complexity (number of prisms) of the vertical decomposition. For $d = 2$, the size of the vertical decomposition of a cell C is proportional to the combinatorial complexity of C, but already for $d = 3$, the size of the vertical decomposition of C can be $\Omega(n^2)$ even when the complexity of C is $O(n)$. A challenging problem is thus to obtain sharp bounds on the complexity of the vertical decomposition of (the cells of) various substructures of $\mathcal{A}(\mathcal{S})$ for $d \geq 3$. Despite extensive work on this problem, see, e.g., [4, 5, 13, 22, 24, 36, 44] for a sample of results, several basic problems remain open. In this paper we settle some of these problems in the affirmative, obtaining sharp bounds on the complexity of the vertical decomposition of various substructures of arrangements, and full arrangements, for $d = 3, 4$; see below for a list of our results. As a major application of these results, we study proximity problems involving lines and points in $\mathbb{R}^3$; see below.

Related work. Collins [29] (see also [20, 43]) had proposed *cylindrical algebraic decomposition* (CAD) as a general technique for decomposing the cells of $\mathcal{A}(\mathcal{S})$ into pseudo-prisms, in any dimension d. However, the number of cells produced is $n^{2^{O(d)}}$. Vertical decomposition can be viewed as an optimized version of CAD, with much smaller complexity. Although vertical decompositions for $d = 2, 3$ have been used since the 1980's [26, 28], Chazelle *et al.* [24] described the construction of vertical decomposition in general, for arrangements of semi-algebraic sets in $\mathbb{R}^d$, and proved a bound of $O^*(n^{2d-3})$ for $d \geq 3$ (where the $O^*(\cdot)$ notation hides subpolynomial factors). They also showed that the vertical decomposition of $\mathcal{A}(\mathcal{S})$ can be computed in $O^*(n^{2d-3})$ expected time. The bound was improved to $O^*(n^{2d-4})$, for $d \geq 4$, by Koltun [36]. These bounds are nearly optimal for $d \leq 4$, and are strongly suspected to be far from optimal for $d \geq 5$. Improving the bound, for $d \geq 5$, is a major 30-years-old open problem in this area (which we do not address in this work).

In many applications, one is interested in computing the vertical decomposition of (the cells of) only a substructure of $\mathcal{A}(\mathcal{S})$. In this case, the goal is to show that if the substructure under consideration has asymptotic complexity $o(n^d)$, then so should be the complexity of its vertical decomposition. This statement is true in the plane, as already mentioned, and has been shown to hold for arrangements of triangles in 3D [22, 47]. Notwithstanding a few results on the vertical decompositions of substructures of 3D and 4D arrangements, see, e.g., [4, 5, 13, 44], the aforementioned fundamental problem has remained largely open for $d \geq 3$. For example, even though the complexity of the union of a set of objects in $\mathbb{R}^3$ in many interesting cases—such as a set of cylinders or a set of fat objects—is known to be $O^*(n^2)$ [15, 18, 30, 31], no subcubic bound was known on the size of the vertical decomposition of the complement of their union. In $\mathbb{R}^4$, the complexity of the lower envelope of n trivariate functions (whose graphs are semi-algebraic sets of constant complexity) is $O^*(n^3)$ (see, e.g., [45]), however, no $o(n^4)$ bound was known on the complexity of the corresponding vertical decomposition of the minimization diagram, which is the xyz-projection of the lower envelope.

We conclude this discussion by noting that special-purpose decomposition schemes have been proposed for decomposing cells in arrangements of hyperplanes, boxes, or simplices, using triangulations, binary space partitions, or variants of vertical decomposition; see, e.g., [14, 16, 19, 33] and references therein. Some of these methods also

work for arrangements of semi-algebraic sets using the so called linearization technique [10], albeit yielding in general much weaker bounds.

Our contributions. The paper contains three sets of main results — (i) sharp bounds on the complexity of vertical decompositions of substructures of arrangements in $\mathbb{R}^3$ and $\mathbb{R}^4$, (ii) efficient algorithms for constructing these decompositions and related structures, and (iii) as a major application domain, efficient data structures for line-point proximity problems in $\mathbb{R}^3$.

Vertical decomposition. We make significant progress on bounding the size of the vertical decomposition of substructures of arrangements in $\mathbb{R}^3$ and $\mathbb{R}^4$, by establishing the following combinatorial bounds.

Union of semi-algebraic sets. Let $\mathcal{S}$ be a family of n semi-algebraic sets of constant complexity in $\mathbb{R}^3$, and let $U(m)$ be an upper bound on the complexity of the union $\mathcal{U}(\mathcal{S}')$ of any subset $\mathcal{S}' \subseteq \mathcal{S}$ of size at most m, for any $m > 0$. (Note that, by definition, $U(m)$ is monotone increasing in m.) We show that the complexity of the vertical decomposition of the complement of $\mathcal{U}(\mathcal{S})$ is $O^*(n^2 + U(n))$ (Section 2).

Lower envelopes. Let $\mathcal{F}$ be a collection of n trivariate functions whose graphs are semi-algebraic sets of constant complexity, and let $\mathcal{A}(\mathcal{F})$ denote the arrangement (in $\mathbb{R}^4$) of their graphs. The *lower envelope* $E_{\mathcal{F}}$ of $\mathcal{F}$ is defined as $E_{\mathcal{F}}(x) = \min_{F\in\mathcal{F}} F(x)$, for $x \in \mathbb{R}^3$. We show that the complexity of the vertical decomposition of the cell of[2] $\mathcal{A}(\mathcal{F})$ lying below (the graph of) $E_{\mathcal{F}}$ is $O^*(n^3)$, thereby matching the general upper bound on the complexity of lower envelopes in $\mathbb{R}^4$ [45] (Section 3).

Sparse arrangements. Let $\mathcal{S}$ be a collection of n semi-algebraic sets of constant complexity in $\mathbb{R}^3$, and let X denote the number of vertices in $\mathcal{A}(\mathcal{S})$. We show that the complexity of the vertical decomposition of the entire arrangement $\mathcal{A}(\mathcal{S})$ is $O^*(n^2 + X)$ (Section 4).

Algorithms. There are a few immediate algorithmic consequences of our combinatorial results:

Computing vertical decompositions. All these vertical decompositions can be constructed, namely, the set of pseudo-prisms in the vertical decomposition can be computed, in time comparable with their respective complexity bounds. Section 5.2 describes the construction for the complement of the union of semi-algebraic sets in $\mathbb{R}^3$, as well as for the lower envelopes (or rather minimization diagrams) of trivariate functions (whose graphs are semi-algebraic sets of constant complexity); the same approach extends to sparse arrangements. We note that Agarwal *et al.* [4] described a randomized algorithm for constructing the vertices, edges, and 2-faces of the minimization diagram of a set of trivariate (constant-complexity semi-algebraic) functions in $O^*(n^3)$ expected time. In addition, with $O^*(n^3)$ preprocessing, their technique can also compute, in $O(\log n)$ time, the function that appears on the lower envelope for a query point $\xi \in \mathbb{R}^3$. (Their algorithm can also compute, in $O^*(n^2 + U(n))$ expected time, the vertices, edges, and 2-faces of the union of a collection $\mathcal{S}$ of semi-algebraic sets in $\mathbb{R}^3$, where $U(m)$, as above, is the maximum complexity of the union of a subset of $\mathcal{S}$ of size m.) However, their algorithm does not compute three-dimensional cells of the minimization diagram, nor does it compute the vertical decomposition of the minimization diagram. See also [13].

Geometric cuttings. Let $\mathcal{S}$ be a collection of n semi-algebraic sets of constant complexity in $\mathbb{R}^d$. Let Π be a substructure of $\mathcal{A}(\mathcal{S})$, defined by a collection of cells of $\mathcal{A}(\mathcal{S})$ that satisfy certain properties (e.g., lying in the complement of the union or lying below the lower envelope). For a parameter $r > 1$, a $(1/r)$-*cutting* of Π (with respect to $\mathcal{S}$) is a set Ξ of pseudo-prisms with pairwise-disjoint relative interiors that cover Π such that the relative interior of each pseudo-prism $\tau \in \Xi$ is crossed by (intersected by but not contained in) at most n/r sets of $\mathcal{S}$. The subset of $\mathcal{S}$ crossed by τ is called the *conflict list* of τ. Our combinatorial results lead to the construction of small-size $(1/r)$-cuttings of Π. Their size is dictated by our new bounds for the complexity of the vertical decomposition of Π. For the case of the complement of the union of sets in $\mathbb{R}^3$, the bound is $O^*(r^2 + U(r))$. For the case of the region below the lower envelope of trivariate functions in $\mathbb{R}^4$, the bound is $O^*(r^3)$. For the case of an entire three-dimensional arrangement of complexity X, we obtain a $(1/r)$-cutting of $\mathcal{A}(\mathcal{S})$, for any parameter $r \le n$, of total complexity $O^*(r^2 + r^3X/n^3)$. The cuttings along with the conflict lists of all of its cells can be constructed in $O(n)$ expected time if r is a constant (Section 5.1).

Point-enclosure queries. Let $\mathcal{S}$ be a family of n semi-algebraic sets in $\mathbb{R}^3$, and let $U(\cdot)$ denote a bound on its union complexity, as above. We obtain a data structure of size and preprocessing cost $O^*(n^2 + U(n))$ that, for a query point $q \in \mathbb{R}^3$, returns all k sets of $\mathcal{S}$ containing q in $O^*(1 + k)$ time. Similarly, for a given family $\mathcal{F}$ of n

[2] Even though this vertical decomposition is in $\mathbb{R}^4$, it is effectively obtained from the vertical decomposition of the *minimization diagram* of $E_{\mathcal{F}}$ in $\mathbb{R}^3$; see below for details.

semi-algebraic trivariate functions, we can construct a data structure of size $O^*(n^3)$ that, for a query point $q \in \mathbb{R}^4$, can report, in $O^*(1+k)$ time, all the k functions of $\mathcal{F}$ whose graphs lie below q. This part is not included in this version, and can be found in the full version [8].

Proximity problems for points and lines in $\mathbb{R}^3$**.** In the third part, building on our vertical-decomposition and geometric-cutting results, we present efficient data structures and algorithms for various proximity problems involving points and lines in $\mathbb{R}^3$.

Nearest line-neighbor to a query point. A set L of n lines in $\mathbb{R}^3$ can be preprocessed, in $O(n \log n)$ expected time, into a data structure of size $O(n)$, so that for a query point $q \in \mathbb{R}^3$, the nearest neighbor of q in L can be returned in $O^*(n^{2/3})$ time (Section 6). We note that a linear-size data structure with $O^*(n^{3/4})$ query time can be obtained by mapping each line of L to a point in $\mathbb{R}^4$ and using four-dimensional semi-algebraic range searching techniques [12]. We also note that a data structure of $O^*(n^3)$ size and $O(\log n)$ query time can be obtained by constructing and preprocessing the Voronoi diagram of the lines in L for point-location queries, following an approach similar to that in [42].

Our data structure constructs a partition tree, as in [10, 46], using geometric cuttings. The main challenge in adapting these preceding approaches to our setting is the construction of a so-called *test set*, namely, a small set of representative queries (typically more involved than the usual queries) so that if the data structure can answer those queries efficiently then it can answer efficiently the query for any point in $\mathbb{R}^3$. Our new results on vertical decomposition of the lower envelope of trivariate functions and on geometric cuttings provide the missing ingredients needed for constructing such test sets. See Section 6 for details.

Nearest point-neighbor to a query line. We can preprocess a set P of n points in $\mathbb{R}^3$, in expected $O^*(n^4)$ time, into a data structure of $O^*(n^4)$ size, so that, for a query line ℓ in $\mathbb{R}^3$, its nearest neighbor in P can be returned in $O(\log n)$ time (Section 7.1). The standard tools would yield a data structure of size $O^*(n^5)$ for answering fast queries.

Roughly speaking, after applying some geometric transformations, we reduce the nearest-neighbor query to a point-location query in a sandwich region enclosed between two envelopes of trivariate functions. As we do not know how to perform this task efficiently in a direct manner, due to the lack of a good bound on the complexity of the vertical decomposition of such a region (see [38], where this is stated as a major open problem), we use a more involved scheme that achieves the desired efficiency.

We note that a linear-size data structure with $O^*(n^{2/3})$ query time can be obtained by using known results on 3D semi-algebraic range searching [12]. Our new results on vertical decomposition of the complement of the union of objects in $\mathbb{R}^3$ leads to a faster solution to a restricted version of this problem. That is, we can preprocess a set of n points in $\mathbb{R}^3$ into a linear-size data structure that returns, in $O^*(n^{1/2})$ time, a point within distance at most 1 from a query line, if there exists one. This problem was recently studied in Agarwal and Ezra [7], and they had obtained a more involved data structure with a similar bound. By combining our vertical-decomposition result with some of their ideas, we obtain a significantly simpler data structure.

Nearest line-neighbor to a query line. We can preprocess a set L of n lines in $\mathbb{R}^3$, in $O^*(n^4)$ expected time, into a data structure of size $O^*(n^4)$, so that the nearest neighbor in L of a query line can be computed in $O(\log n)$ time (Section 7.2). Again, we note that a linear-size data structure with $O^*(n^{3/4})$ query time can be obtained by using standard four-dimensional semi-algebraic range searching techniques [12], and that a structure of size $O^*(n^5)$ for the fast query regime can also be obtained by standard methods.

Offline nearest-neighbor queries. Let us now consider the case when all queries are given in advance. That is, we have a set L of n lines and a set P of m points in $\mathbb{R}^3$, and the goal is to compute the nearest neighbor in P of each line of L. We present a randomized algorithm with $O^*(m^{4/7}n^{6/7}+m+n)$ expected running time. We note that by plugging our on-line algorithm with the standard space/query-time trade-off techniques would lead to an algorithm with the inferior $O^*(m^{8/11}n^{9/11}+m+n)$ expected running time. We have also studied the bichromatic closest pair variant, where we wish to compute the closest line-point pair in $L \times P$. As it turns out, this problem can be solved faster, in expected time $O^*(m^{3/5}n^{4/5}+m+n)$. This part is not included in this version, and can be found in the full version [8].

2 Vertical Decomposition of the Complement of the Union

Let $\mathcal{S}$ be a collection of n semi-algebraic sets of constant complexity in $\mathbb{R}^3$. For any subset $\mathcal{S}'$ of $\mathcal{S}$, let $\mathcal{U}(\mathcal{S}')$ denote the union of $\mathcal{S}'$, and let $\mathcal{C}(\mathcal{S}')$ denote the complement of $\mathcal{U}(\mathcal{S}')$. Let $U(m)$ denote the maximum complexity

of $\mathcal{U}(S')$—namely, the number of vertices, edges and 2-faces of the union boundary—over all subsets $\mathcal{S}'$ of size at most m. Clearly $U(m) = O(m^3)$, but as mentioned in the introduction, $U(m) = O^*(m^2)$ in many interesting cases. Let $\mathrm{VD}(\mathcal{S})$ denote the *vertical decomposition* of $\mathcal{C} = \mathcal{C}(\mathcal{S})$, and let $C(n)$ denote the maximum complexity of $\mathrm{VD}(\mathcal{S})$, where the maximum is taken over all collections of n semi-algebraic sets of constant complexity. Our goal is to obtain a sharp bound on $C(n)$.

A pair (e, e') of edges of $\mathcal{A}(\mathcal{S})$ is called *vertically visible* if there exists a vertical line λ that meets both e and e', so that the relative interior of the segment of λ connecting e and e' does not meet the boundary of any set of $\mathcal{S}$, and we refer to the pair of points $(\lambda \cap e, \lambda \cap e')$ as a *vertical visibility*. A pair (e, e') of edges can give rise to more than one but at most $O(1)$ vertical visibilities. It is well known (see, e.g., [45]) that $C(n)$ is proportional to $U(n)$ plus the number of vertical visibilities between pairs of edges of $\partial\mathcal{U}$ that occur within $\mathcal{C}$, so it suffices to bound the latter quantity.

To bound the number of vertical visibilities, we fix an edge e of $\partial\mathcal{U}$, regarding e as the lower edge in the vertical visibilities that we seek,[3] and erect a *vertical curtain* $V(e)$ over e, which is the (two-dimensional) union of all z-vertical rays emanating upwards from the points of e. The boundary of each set $S \in \mathcal{S}$ (ignoring the two that form e) intersects $V(e)$ in a one-dimensional curve γ_S, which can be empty or disconnected, but is of constant complexity. Note that none of the curves γ_S cross e, for such an intersection would be a vertex of the arrangement of $\mathcal{S}$ and, by definition, e cannot contain such a vertex.

We form the lower envelope E_e of the curves γ_S, and note that each breakpoint a of E_e, at which two curves meet, lies on some edge e' of $\partial\mathcal{U}$ which forms a vertically visible pair with e, with the vertical visibility taking place between a and e. The other breakpoints, formed at endpoints of connected portions of the curves, occur when a vertical line (supporting a ray of the curtain $V(e)$) is tangent to some $S \in \mathcal{S}$; that is, the breakpoint occurs on the vertical silhouette of S. It is easy to show that the overall number of vertical visibilities involving silhouettes is only $O^*(n^2)$. Indeed, there are $O(n)$ silhouettes, each of constant complexity, and the vertical visibilities that they are involved in correspond to breakpoints of lower or upper envelopes within the vertical curtains that they span. As each envelope can be regarded as the lower envelope of univariate functions, it has $O^*(n)$ complexity [45], and the claim follows.

To facilitate the forthcoming analysis, we turn the problem into a bipartite problem, where each set of $\mathcal{S}$ is assigned at random a color red or blue, yielding a partition $\mathcal{S} = \mathcal{R} \cup \mathcal{B}$, where $\mathcal{R}$ (resp., $\mathcal{B}$) is the set of all red (resp., blue) sets, and our goal is to bound the number of vertical visibilities between red-red edges (edges formed by the intersection of the boundaries of two red sets) and blue-blue edges (those formed by the intersection of the boundaries of two blue sets). Note that a red-red edge e on the boundary of the union of $\mathcal{R}$ is not necessarily an original edge of the boundary of $\mathcal{U}(\mathcal{S})$, as e may contain red-red-blue vertices (or even be fully contained in a blue set). Still, if there exists a vertical visibility in $\mathcal{C}(\mathcal{S})$ whose lower endpoint b lies on e, then b lies on a portion of e that forms an edge of $\partial\mathcal{U}(\mathcal{S})$. Of course, not all vertically visible pairs are captured in this coloring scheme. Nevertheless, it is easily checked that the expected number of visible pairs with this coloring is $1/8$ of the overall number of visible pairs, so, up to this factor, there is no loss of generality in using this coloring scheme.

So the setup that we face is: We are given a set $\mathcal{R}$ of m red sets and a set $\mathcal{B}$ of n blue sets (in the above scheme, both m and n are half the size of $\mathcal{S}$ in expectation), and our goal is to bound the number $C(m, n)$ of vertical visibilities between pairs (e, e') of edges, where e is a red-red edge and e' is a blue-blue edge, and the vertical visibility takes place in the complement of $\mathcal{U}(\mathcal{R} \cup \mathcal{B})$.

We estimate $C(m, n)$ using an extension of the recursive analysis in [38, Section 2].[4] We fix some sufficiently large constant parameter k, and partition $\mathcal{B}$ arbitrarily into k subsets $\mathcal{B}_1, \ldots, \mathcal{B}_k$, each of size n/k (ignoring rounding issues). We solve the problem recursively for $\mathcal{R}$ and each $\mathcal{B}_i$. Each subproblem yields at most $C(m, n/k)$ vertical visibilities. Note that these vertical visibilities are not necessarily vertical visibilities in the full red-blue setup, because sets in other subsets $\mathcal{B}_j$ may show up between the edges in such a pair and destroy the vertical visibility between them. Nevertheless, each original vertical visibility is either one of these recursively obtained visibilities, or arises at a pair (e, e') where e is a red-red edge and e' is a blue-blue edge formed by the intersection of two boundaries of sets in different subsets $\mathcal{B}_i$, $\mathcal{B}_j$. We now proceed to bound the number of pairs of the latter kind.

[3] We assume that the two sets whose boundaries intersect at e lie locally below e, for otherwise e cannot play the role of the bottom edge of a vertically visible pair in the complement of the union.

[4] We credit this work for providing us the initial inspiration that their technique can be adapted to apply in our settings too.

To do so, fix a red-red edge e, and assume that e plays the role of the bottom edge in a vertically visible pair. Consider the upward vertical curtain $V(e)$ of e, and form within $V(e)$ the k blue envelopes $E_e^{(1)}, \ldots, E_e^{(k)}$, where $E_e^{(i)}$ is the lower envelope of the curves γ_S, for $S \in \mathcal{B}_i$, for $i = 1, \ldots, k$. The breakpoints of the envelopes (ignoring silhouette breakpoints) correspond to recursively obtained pairs (e, e') (as noted, not all breakpoints yield visibilities in the full setup), but we are also interested in the additional breakpoints of the overall lower envelope E_e of these k envelopes.

Let $M_e^{(i)}$ denote the number of breakpoints of $E_e^{(i)}$, for $i = 1, \ldots, k$, and put $M_e = \sum_i M_e^{(i)}$. Notice that $\sum_e M_e^{(i)}$ is the number of vertical visibilities between $\mathcal{R}$ and $\mathcal{B}_i$, so it is at most $C(m, n/k)$. Thus $\sum_e M_e \le kC(m, n/k)$.

Inspired by the analysis in [38], we follow a technique similar to one used by Har-Peled [34] in a different context. Specifically, we partition $V(e)$ into vertical sub-curtains $V_1(e), \ldots, V_t(e)$ by upward vertical rays, so that the overall number of breakpoints of the individual envelopes within each sub-curtain is k, except possibly for the last sub-curtain, where the number is at most k, so $t \le 1 + M_e/k$. Within each sub-curtain $V_j(e)$ there are only at most $2k$ blue curves γ_S that participate in the envelopes $E_e^{(i)}$, of which k show up on the envelopes at an extreme ray of $V_j(e)$, and at most k others replace them along the various envelopes, within the sub-curtain. Hence, within any fixed $V_j(e)$, E_e is the lower envelope of at most $2k$ connected subarcs of boundary curves γ_S, so its combinatorial complexity is at most $\lambda_s(2k)$, where $\lambda_s(m)$ is the near-linear maximum length of Davenport-Schinzel sequences of order s on m symbols, for some constant parameter s that depends on the complexity of the sets of $\mathcal{S}$ [45]. We write this bound as $k\beta(k)$, for an appropriate near-constant extremely slowly growing function $\beta(k)$, and conclude that the number of breakpoints of E_e within each sub-curtain is at most $k\beta(k)$, for a total of at most $kt\beta(k) = (k + M_e)\beta(k)$ breakpoints. Summing over all red-red edges e, we obtain

$$C(m, n) \le \left(\sum_e (k + M_e) \right) \beta(k) \le k\beta(k)C(m, n/k) + k\beta(k)U(m).$$

We next switch the roles of red and blue, and apply the same analysis to each pair $\mathcal{R}$, $\mathcal{B}_i$ of sets, keeping $\mathcal{B}_i$ fixed and partitioning $\mathcal{R}$ into k subsets of size m/k each. (We now reverse the direction of the z-axis, considering downward-directed vertical curtains erected from the edges formed by the sets of $\mathcal{B}_i$.) The analysis proceeds more or less verbatim, and yields the following bound on the number of vertical visibilities:

$$C(m, n) \le k^2\beta^2(k)C(m/k, n/k) + k^2\beta^2(k)U(n/k) + k\beta(k)U(m)$$

If $U(m) = O^*(m^2)$, we obtain the recurrence

$$C(m, n) \le k^2\beta^2(k)C(m/k, n/k) + k\beta(k)O^*(m^2) + \beta^2(k)O^*(n^2).$$

Note that the right-hand side of this recurrence also subsumes the number of $O^*(m^2 + n^2)$ vertical visibilities that involve the silhouettes of the red and blue sets.

We solve this recurrence for the original setup, where m and n are both roughly half the total number of sets, which we continue to denote by n, with some abuse of notation. By choosing k to be a sufficiently large constant, the solution of the resulting recurrence is $O^*(n^2)$. We thus conclude that the number of vertical visibilities between pairs of edges of $\mathcal{U}(\mathcal{S})$ is $O^*(n^2)$. A similar analysis applies when $U(n)$ is superquadratic. In this case the bound on the complexity of the vertical decomposition is $O^*(U(n))$, as is easily checked. Putting everything together, we obtain the following main result of this section.

THEOREM 2.1. *Let $\mathcal{S}$ be a collection of n constant-complexity semi-algebraic sets in $\mathbb{R}^3$, with an upper bound $U(m)$ on the combinatorial complexity of the union of any subset of $\mathcal{S}$ of size m. Then the size of the vertical decomposition of the complement of the union of $\mathcal{S}$ is $O^*(n^2 + U(n))$.*

3 Vertical Decomposition of Lower Envelopes in $\mathbb{R}^4$

Let $\mathcal{F}$ be a collection of n trivariate semi-algebraic functions of constant complexity, let $E = E_{\mathcal{F}}$ denote the lower envelope of $\mathcal{F}$, let $E^- = E_{\mathcal{F}}^-$ denote the portion of $\mathbb{R}^4$ below E, and let $M = M_{\mathcal{F}}$ denote the minimization diagram of E, namely the projection of E onto the xyz-space. Our goal is to estimate the combinatorial complexity of the

vertical decomposition of M. This three-dimensional decomposition can then be lifted up in the w-direction to induce a suitable decomposition of E^-, which we refer to as the vertical decomposition of E. We note that the complexity of (the undecomposed) E and of M is $O^*(n^3)$ [45]. The main result of this section yields the same asymptotic bound for their vertical decomposition:

THEOREM 3.1. *The complexity of the vertical decomposition of the lower envelope (that is, of the minimization diagram) of a collection of n trivariate semi-algebraic functions of constant complexity is $O^*(n^3)$.*

Proof. We assume that the functions of $\mathcal{F}$ are in general position, continuous and totally defined. None of these assumptions are essential, but they simplify the analysis. We identify each function of $\mathcal{F}$ with its three-dimensional graph. We recall the way in which the vertical decomposition VD of M is constructed. We fix a function a in $\mathcal{F}$. For each function $b \in \mathcal{F} \setminus \{a\}$, we use $\sigma_{ab} = \sigma_{ba}$ to denote the xyz-projection of the two-dimensional intersection surface $a \cap b$. The surface σ_{ab} partitions the xyz-space into the regions σ^+_{ab} and σ^-_{ab}, where σ^+_{ab} (resp., σ^-_{ab}) consists or those points (x, y, z) for which $a(x, y, z) \geq b(x, y, z)$ (resp., $a(x, y, z) \leq b(x, y, z)$). We observe that the complement $\mathcal{C}_a$ of the union $\mathcal{U}_a := \bigcup \{\sigma^+_{ab} \mid b \in \mathcal{F} \setminus \{a\}\}$ is precisely the portion of the xyz-space over which a attains the envelope E.

We now construct the three-dimensional vertical decomposition, denoted as VD_a, of $\mathcal{C}_a$, and repeat this construction to each complement $\mathcal{C}_a$, over $a \in \mathcal{F}$, observing that the regions $\mathcal{C}_a$ are pairwise openly disjoint. The union of all these decompositions yields the vertical decomposition of $M_{\mathcal{F}}$, and, as mentioned above, the vertical decomposition of $E_{\mathcal{F}}$ is obtained by lifting this decomposition to $E_{\mathcal{F}}$ (or to $E^-_{\mathcal{F}}$, see below), in a straightforward manner.

We comment that, as already noted, we can also obtain by this approach the vertical decomposition of E^-. Each cell τ in the decomposition of M is lifted to the semi-unbounded region

$$\{(x, y, z, w) \mid (x, y, z) \in \tau \text{ and } w \leq E(x, y, z)\}.$$

We have thus (almost) reduced the problem to that studied in Section 2. The difference is that there we assumed that the complexity of the union of any subcollection of at most m of the given objects is $O^*(m^2)$, or at least that we have some (subcubic) bound $U(m)$ on that complexity. Here, though, this no longer holds. That is, considering the entire collection $\mathcal{F}$, and denoting by M_a the complexity of $\mathcal{U}_a$, all we know is that $\sum_a M_a = O^*(n^3)$, so we have the bound $O^*(n^2)$ only for the *average* value of M_a. To overcome this technicality, we modify the previous analysis as follows.

Recall that in Section 2 we have reduced the problem to a bichromatic problem by assigning to each object the color red or blue at random. Here we extend this technique to obtain a trichromatic reduction, by assigning to each function the color red, blue or green at random. We now consider only unions $\mathcal{U}_a$ for green functions a, and within the complement $\mathcal{C}_a$ of any of these unions, we only consider vertical visibilities between red-red edges and blue-blue edges (technically, they are green-red-red and green-blue-blue edges), exactly as in Section 2. Again, any vertical visibility that arises in the original decomposition has a constant probability to show up as a green-red-red vs. green-blue-blue visibility in the trichromatic version.

For each green function a, the overhead terms that appear in the analysis can be written as $M(\{a\}, \mathcal{R}, \mathcal{B})$ and $M(\{a\}, \mathcal{R}, \mathcal{B}_i)$, where, for arbitrary sets $\mathcal{G}$, $\mathcal{R}$, $\mathcal{B}$ of green, red, and blue objects, respectively, $M(\mathcal{G}, \mathcal{R}, \mathcal{B})$ denotes the number of the green-red-red and green-blue-blue edges of the undecomposed envelope of $\mathcal{G} \cup \mathcal{R} \cup \mathcal{B}$. Here $\mathcal{R}$, $\mathcal{B}$, and the $\mathcal{B}_i$'s may be recursively obtained subsets of the original sets. Summing these quantities over a, we obtain $M(\mathcal{G}, \mathcal{R}, \mathcal{B})$ and $M(\mathcal{G}, \mathcal{R}, \mathcal{B}_i)$, respectively. We also use the notation $M(u, v, w)$ to denote the maximum value of $M(\mathcal{G}, \mathcal{R}, \mathcal{B})$ for $|\mathcal{G}| \leq u$, $|\mathcal{R}| \leq v$ and $|\mathcal{B}| \leq w$.

Consider, say, a green-red-red edge e that appears on the boundary of (the complement $\mathcal{C}_a$ of) the union $\mathcal{U}_a$ for some green function a (the same argument holds for green-blue-blue edges). If we replace $\mathcal{G}$ by a subset $\mathcal{G}'$ that contains a, $\mathcal{C}_a$ can only grow, since fewer regions σ^+_{ab} form the union $\mathcal{U}_a$. Hence e does not disappear, and can only extend, possibly even merge with other edges formed by the same triple of functions. In particular, the number of vertical visibilities in $\mathcal{C}_a$ between green-red-red edges and green-blue-blue edges can only increase.

We use this observation as follows. In the first two-step recursive round, as described in Section 2, we first partition $\mathcal{G}$ into k subsets $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_k$, each of size n/k, apply the analysis to each $\mathcal{G}_i$ and $\mathcal{R}$ and $\mathcal{B}$, and then sum up the resulting bounds for $i = 1, \ldots, k$. Denote by $C(u, v, w)$ the maximum number of vertical visibilities for sets of at most u green, v red, and w blue functions. The overhead term will be at most $O(M(u, v, w)) = O^*((u+v+w)^3)$,

and the recursive term will be at most $C(u/k, v/k, w/k)$ at each recursive subproblem. Therefore, by applying the recursive relation from Section 2 on the number of red-blue vertical visibilities, we obtain the recurrence:

$$C(u,v,w) \leq \sum_{i=1}^{k} k^2\beta^2(k)C(u/k, v/k, w/k) + k\beta(k)M(u/k, v, w),$$

which leads to the recursive relation:

$$C(u,v,w) \leq k^3\beta^2(k)C(u/k, v/k, w/k) + k^2\beta(k)O^*((u+v+w)^3).$$

The recurrence terminates when one of $u, v, w \leq k$. It can be verified that $C(u,v,w) = O^*((u+v+w)^3)$. It then follows that $C(u,v,w) = O^*((u+v+w)^3)$ for any values of u, v, w, and this completes the proof of Theorem 3.1. □

4 Vertical Decomposition of Arrangements in $\mathbb{R}^3$

Let $\mathcal{S}$ be a set of n surfaces or surface patches in $\mathbb{R}^3$ in general position, each of which is semi-algebraic of constant complexity, and let X denote the number of vertices of $\mathcal{A}(\mathcal{S})$. For simplicity, and with no loss of generality, we assume that the surfaces are graphs of possibly partially defined continuous functions. This can be ensured by cutting surfaces into surface patches at their silhouettes and at their curves of singularity. We show that the complexity of the vertical decomposition of $\mathcal{A}(\mathcal{S})$ is $O^*(n^2 + X)$.

As in Section 2, it suffices to bound the number of vertical visibilities between pairs of edges of $\mathcal{A}(\mathcal{S})$. Again, we randomly color each surface as either red or blue, and only consider visibilities between red-red edges and blue-blue edges, in which the red-red edge lies below the blue-blue edge. An original vertical visibility has 1/8 probability to appear as a visibility of the desired kind under the coloring scheme. That is, up to a constant factor, the bound that we seek is also an upper bound for the original uncolored case. Here too, each monochromatic edge e may in general be the union of several original edges of $\mathcal{A}(\mathcal{S})$. Therefore the number of these monochromatic edges is at most $O(X)$. As before, we denote the subsets of red surfaces and blue surfaces as $\mathcal{R}$ and $\mathcal{B}$, respectively, and put $m := |\mathcal{R}|$, $n := |\mathcal{B}|$, slightly abusing the notation, as above.

The high-level analysis proceeds more or less as in Section 2. That is, we apply a two-step partitioning scheme, in which we first partition the blue surfaces into k subsets $\mathcal{B}_1, \ldots, \mathcal{B}_k$, each of n/k surfaces (in fact, the number of these surfaces in each subcell is at most $2n/k$—see below for the details of the analysis). Then, for each red-red edge e, we form k separate lower envelopes of the blue surfaces, one for each $\mathcal{B}_i$, within the (upward) vertical curtain erected from e, and analyze the complexity of the lower envelope of all these envelopes.

Denote by $C(m, n, X_1, X_2)$ the maximum number of vertical visibilities between red-red edges and blue-blue edges in an arrangement of a set $\mathcal{R}$ of at most m red surfaces and a set $\mathcal{B}$ of at most n blue surfaces, so that the complexity (number of vertices) of $\mathcal{A}(\mathcal{R})$ is at most X_1 and the complexity of $\mathcal{A}(\mathcal{B})$ is at most X_2. Observe that $X_1 + X_2 \leq X$.

A major new aspect of the analysis is in handling the parameter X, now replaced by X_1 and X_2. The issue is that we have no control on how X_1 and X_2 are distributed over the subproblems that arise when we partition $\mathcal{B}$ into k arbitrary subsets, and then do the same for $\mathcal{R}$, as we did in Section 2.

We overcome this issue by partitioning each of $\mathcal{R}$, $\mathcal{B}$ into k *random* subsets, say by choosing the subset to which a surface belongs independently and uniformly at random. Specifically, consider the first partitioning step, where $\mathcal{B}$ is split. We form a random partition of $\mathcal{B}$ into k subsets $\mathcal{B}_1, \ldots, \mathcal{B}_k$, where a surface $\sigma \in \mathcal{B}$ is assigned to a subset $\mathcal{B}_i$, $1 \leq i \leq k$, which is chosen with probability $1/k$, independent of the assignment of the remaining surfaces in $\mathcal{B}$. This probabilistic model obeys the multinomial distribution with k "categories". In particular, this implies that the size of each $\mathcal{B}_i$ is a binomial random variable with parameters n and $1/k$. Similarly, when we apply such a random partition to $\mathcal{R}$ at the second partitioning step, we obtain a partition into k subsets $\mathcal{R}_1, \ldots, \mathcal{R}_k$, where the size of each $\mathcal{R}_j$ is a binomially distributed random variable with parameters m and $1/k$. We clearly have $E[|\mathcal{B}_i|] = n/k$, for each $1 \leq i \leq k$, and $E[|\mathcal{R}_j|] = m/k$, for each $1 \leq j \leq k$.

Using standard probabilistic arguments, exploiting the multiplicative Chernoff bound [17], we conclude that,

with high probability,

$$|\mathcal{B}_i| \leq n/k \left(1 + O\left(\sqrt{\frac{k}{n}\log n}\right)\right), \qquad \text{for each } 1 \leq i \leq k, \text{ and}$$
$$|\mathcal{R}_j| \leq m/k \left(1 + O\left(\sqrt{\frac{k}{m}\log m}\right)\right), \qquad \text{for each } 1 \leq j \leq k.$$

By choosing k appropriately, we can assume that, with high probability, these upper bounds do not exceed $2n/k$, and $2m/k$, respectively.

Moreover, at the first partitioning step, a blue-blue edge e' is assigned to a specific subset $\mathcal{B}_i$ with probability at most $1/k^2$ (here too, a blue-blue edge of $\mathcal{A}(\mathcal{B}_i)$ may be the union of several original edges of $\mathcal{A}(\mathcal{B})$). Specifically, e' is defined by at most four surfaces. That is, if e' contains two endpoints (each of which is a vertex of the arrangement obtained by the intersection of a triple of surfaces) then this number is four, if it has only one endpoint then e' is defined by three surfaces, otherwise, it is defined by a pair of surfaces (recall that we exclude silhouette and singularity edges, in which case there is only a single surface defining an edge).

In the first two scenarios $\mathcal{B}_i$ has to contain the triple of surfaces defining an endpoint of e' (or the quadruple defining both endpoints), which occurs with probability at most $1/k^3$. In the latter scenario the pair of surfaces defining e' has to be assigned to $\mathcal{B}_i$, which happens with probability $1/k^2$. Therefore the expected complexity of the arrangement $\mathcal{A}(\mathcal{B}_i)$ is $O(n^2/k^2 + X_2/k^3)$. We comment that the events that edges show up in a specific subset are not independent. However, we claim below that, with high probability, the complexity of $\mathcal{A}(\mathcal{B}_i)$ is $O(n^2/k^2 + X_2/k^2)$, for each $1 \leq i \leq k$. This bound is slightly worse than the expected complexity, but it suffices for the analysis to proceed.

Indeed, since we have, with high probability, $|\mathcal{B}_i| \leq 2n/k$, for each $1 \leq i \leq k$, we immediately conclude that the number of edges of $\mathcal{A}(\mathcal{B}_i)$ that are formed by pairs of surfaces is $O(n^2/k^2)$ (with high probability). Regarding the number of edges that are formed by a triple (or a quadruple) of surfaces, their expected number Y is $O(X_2/k^3)$, as observed above. Using Markov's inequality we conclude that the probability that the actual number of such edges exceeds $2kY$ is at most $1/(2k)$. That is, with probability at least $1 - 1/(2k)$, the number of such edges in $\mathcal{A}(\mathcal{B}_i)$ is at most $O(X_2/k^2)$. Using the probability union bound, we obtain that this bound holds for all sets B_i, $1 \leq i \leq k$, with probability at least $1/2$. We comment that this event is conditioned on the event that $|\mathcal{B}_i| \leq 2n/k$, for each $1 \leq i \leq k$ (which occurs with very high probability), so using the rule of conditional probability, we can assume that with probability at least $1/4$ the overall complexity of $\mathcal{A}(\mathcal{B}_i)$ is at most $O(n^2/k^2 + X_2/k^2)$, for each $1 \leq i \leq k$. By the probabilistic method [17] this implies that there exists such a partition $\mathcal{B}_1, \ldots, \mathcal{B}_k$.

Hence, a suitable adaptation of the analysis in Section 2 yields the first-level recurrence (where $c > 0$ below is an absolute constant):

$$C(m, n, X_1, X_2) \leq k\beta(k)C(m, 2n/k, X_1, c(n^2/k^2 + X_2/k^2)) + k\beta(k)X_1 + O^*(mn),$$

for a suitable near-constant extremely slowly growing function $\beta(k)$. The overhead term $O^*(mn)$ comes from vertical visibilities that involve silhouettes and singularities, and follows by an argument similar to that in Section 2.

We next switch the roles of red and blue, and apply the same analysis to each pair $\mathcal{R}$, $\mathcal{B}_i$ of surfaces, keeping $\mathcal{B}_i$ fixed and partitioning $\mathcal{R}$ into k random subsets, as above, each of which is of size at most $2m/k$ (with high probability). The analysis proceeds in a similar manner, and yields the bound

$$k^2\beta^2(k)C(2m/k, 2n/k, c(m^2/k^2 + X_1/k^2), c(n^2/k^2 + X_2/k^2)) + O_k(X_1 + X_2) + O_k^*(mn)$$

on the number of vertical visibilities, where the $O_k(\cdot)$ notation indicates that the constant of proportionality depends on k. That is, we obtain the recurrence

$$C(m, n, X_1, X_2) \leq k^2\beta^2(k)C(2m/k, 2n/k, c(m^2/k^2 + X_1/k^2), c(n^2/k^2 + X_2/k^2)) + O_k(X_1 + X_2) + O_k^*(mn)$$

By choosing k to be a sufficiently large constant, the solution of the recurrence is easily seen to be

$$C(m, n, X_1, X_2) = O^*(m^2 + n^2 + X_1 + X_2).$$

That is, replacing m and n by the original value of n, and X_1, X_2 by the original quantity X, we obtain the following:

THEOREM 4.1. *Let $\mathcal{S}$ be a collection of n constant-complexity semi-algebraic surfaces or surface patches in $\mathbb{R}^3$, and let X be the number of vertices in $\mathcal{A}(\mathcal{S})$. Then the complexity of the vertical decomposition of $\mathcal{A}(\mathcal{S})$ is $O^*(n^2+X)$.*

5 Constructing Cuttings and Decompositions

5.1 Constructing cuttings Let $\mathcal{S}$ be a collection of n semi-algebraic sets of constant complexity in $\mathbb{R}^d$. Let Π be a substructure of $\mathcal{A}(\mathcal{S})$, say, defined by a collection of cells of $\mathcal{A}(\mathcal{S})$ that satisfy certain properties (e.g., lying in the complement of the union or lying below the lower envelope). For a parameter $r>1$, a $(1/r)$-*cutting* of Π is a set Ξ of pseudo-prisms with pairwise-disjoint relative interiors that cover Π, such that the relative interior of each pseudo-prism $\tau\in\Xi$ is crossed by (intersected by but not contained in) at most n/r sets of $\mathcal{S}$. The subset of $\mathcal{S}$ crossed by τ is called the *conflict list* of τ.

It is well known that the random-sampling paradigm can be used to construct a $(1/r)$-cutting [1, 23, 35, 39]. Namely, set $s=cr\log r$, where c is a sufficiently large constant. Let $\mathcal{R}\subseteq\mathcal{S}$ be a random subset of $\mathcal{S}$ of size s, and let $\mathrm{VD}(\mathcal{R})$ be the vertical decomposition of $\mathcal{A}(\mathcal{R})$. For each cell $\tau\in\mathrm{VD}(\mathcal{R})$, let $\mathcal{S}_\tau\subset\mathcal{S}$ be the subset of $\mathcal{S}$ that crosses τ. By construction, $\mathcal{S}_\tau\cap\mathcal{R}=\emptyset$ and τ is a semi-algebraic set of constant complexity, therefore using a standard random-sampling argument [27, 35], it can be shown that $|\mathcal{S}_\tau|\le n/r$ for all $\tau\in\mathrm{VD}(\mathcal{R})$ with probability at least $1/2$ assuming the constant c is chosen sufficiently large. Therefore, to construct a $(1/r)$-cutting Ξ of Π, we only have to decide which of the cells of $\mathrm{VD}(\mathcal{R})$ should be included in Ξ to ensure that they cover Π.

If $\mathcal{S}$ is a set of semi-algebraic sets in $\mathbb{R}^3$ and we wish to compute a $(1/r)$-cutting of $\mathcal{C}(\mathcal{S})$, the complement of the union of $\mathcal{S}$, we set $\Xi=\{\tau\in\mathrm{VD}(\mathcal{R})\mid\tau\subseteq\mathcal{C}(\mathcal{R})\}$. Since $\mathcal{R}\subseteq\mathcal{S}$, $\mathcal{C}(\mathcal{S})\subseteq\mathcal{C}(\mathcal{R})$, and thus Ξ is guaranteed to cover $\mathcal{C}(\mathcal{S})$. By Theorem 2.1, $|\Xi|=O^*(r^2+U(r))$. In contrast, if we want to construct a $(1/r)$-cutting of the entire $\mathcal{A}(\mathcal{S})$, we set $\Xi=\mathrm{VD}(\mathcal{R})$. If $\mathcal{A}(\mathcal{S})$ has X vertices, then the expected number of vertices in $\mathcal{A}(\mathcal{R})$ is $O(r^2+Xr^3/n^3)$, and thus, by Theorem 4.1, the expected size of Ξ is $O^*(r^2+Xr^3/n^3)$. (If the size of Ξ is more than twice its expected size, we discard Ξ and repeat the construction.) Finally, if $\mathcal{S}$ represents graphs of a set of trivariate functions in $\mathbb{R}^4$ and we wish to construct a $(1/r)$-cutting of the portion of $\mathcal{A}(\mathcal{S})$ lying below the lower envelope of $\mathcal{S}$, we set Ξ to be the set of cells of $\mathrm{VD}(\mathcal{R})$ that lie below the lower envelope of $\mathcal{R}$. By Theorem 3.1, $|\Xi|=O^*(r^3)$. Hence, we conclude the following:[5]

THEOREM 5.1. *(i) Let $\mathcal{S}$ be a collection of n semi-algebraic sets of constant complexity in $\mathbb{R}^3$, and let $U(m)$ be an upper bound on the complexity of the union of at most m objects of $\mathcal{S}$. There exists a $(1/r)$-cutting of $\mathcal{C}(\mathcal{S})$, the complement of the union of $\mathcal{S}$, of size $O^*(r^2+U(r))$.*

(ii) Let $\mathcal{F}$ be a collection of n trivariate semi-algebraic functions of constant complexity. There exists a $(1/r)$-cutting of the region below the lower envelope of $\mathcal{F}$ of size $O^*(r^3)$.

(iii) Let $\mathcal{S}$ be a collection of n constant-complexity semi-algebraic surfaces or surface patches in $\mathbb{R}^3$, so that the number of vertices in $\mathcal{A}(\mathcal{S})$ is X. Then there exists a $(1/r)$-cutting for $\mathcal{S}$ of size $O^*(r^2+r^3X/n^3)$.

For contestant values of r, these cuttings, along with the conflict lists of their cells, can be computed in $O(n)$ expected time (where the constant of proportionality depends on r).

5.2 Constructing vertical decompositions We now describe algorithms for constructing vertical decompositions for the cases studied in Sections 2–4.

Complements of unions in $\mathbb{R}^3$. Let $\mathcal{S}$ be a collection of n semi-algebraic sets (each of constant complexity) in $\mathbb{R}^3$ such that the maximum complexity of the union of any subset $\mathcal{S}'$ of $\mathcal{S}$ of at most $m\le n$ sets is $U(m)$. Let $\mathcal{C}(\mathcal{S}')$ denote the complement of $\mathcal{U}(\mathcal{S}')$.

We present below an algorithm that constructs, in $O^*(n^2+U(n))$ expected time, the vertical decomposition of $\mathcal{C}(\mathcal{S})$. More precisely, it constructs the set of pseudo-prisms in the vertical decomposition of $\mathcal{C}(\mathcal{S})$. As a main step of the algorithm, we perform the subtask of reporting all the vertical visibilities, within $\mathcal{C}(\mathcal{S})$, between pairs of edges (e,e') that lie on $\partial\mathcal{U}(\mathcal{S})$. By Theorem 2.1, the number of these vertical visibilities is $O^*(n^2+U(n))$. We also compute the vertices, edges, and 2-faces of $\mathcal{U}(\mathcal{S})$ in $O^*(n^2+U(n))$ expected time, e.g., using the randomized incremental algorithm described in [4]. Then the pseudo-prisms in $\mathrm{VD}(\mathcal{S})$ can be computed in a fairly standard

[5] It is possible to reduce the size of the cuttings by a polylogarithmic factor using a two-level sampling scheme as described in [11, 23, 25, 39]. Since we are using $O^*()$ notation and are ignoring subpolynomial factors, we described a simpler, albeit slightly weaker, construction.

(though somewhat tedious) manner by traversing all the faces and edges of $\partial\mathcal{U}(\mathcal{S})$ and tracking their vertical visibilities. We omit the details from here in the interest of brevity, and refer the reader to [22], where a similar method was used for computing the vertical decomposition of an arrangement of triangles in $\mathbb{R}^3$.

We follow a randomized divide-and-conquer scheme to compute vertical visibilities. Let $1 \leq r \leq n$ be a sufficiently large constant parameter. If $|\mathcal{S}| \leq n_0$, where n_0 is a constant that depends on r, we report all pairs of vertical visibilities between the edges on $\partial\mathcal{C}(\mathcal{S})$ in a brute-force manner. Otherwise, we recursively construct a $(1/(2r))$-cutting Ξ of $\mathcal{C}(\mathcal{S})$ of size $O^*(r^2 + U(r))$, using Theorem 5.1 (i). (We comment that the actual reporting is done only at the bottom of the recurrence.) For each cell $\tau \in \Xi$, let $\mathcal{S}_\tau \subset \mathcal{S}$ be its conflict list, the family of input sets that cross the relative interior of τ, plus the $O(1)$ input sets that define the cell τ. By construction, $|\mathcal{S}_\tau| \leq n/(2r) + O(1) \leq n/r$. As is easily verified, any edge pair (e, e') (that lie on $\partial\mathcal{C}(\mathcal{S})$) of vertical visibility within $\mathcal{C}(\mathcal{S})$ must be reported during this process, since the vertical segment ρ connecting e and e' must be contained in some prism cell of Ξ. Otherwise, this would imply that one of the input sets crosses ρ, but this violates the definition of vertical visibility. The overall expected running time $T(n)$ to report all pairs of vertical visibility obeys the recurrence:

$$T(n) = O^*(r^2 + U(r))T(n/r) + O^*(n),$$

where the overhead term accounts for computing Ξ and the conflict lists of all the cells of Ξ. Using induction, it can be verified that the solution is $T(n) = O^*(n^2 + U(n))$. We have thus shown:

THEOREM 5.2. *Let $\mathcal{S}$ be a collection of n constant-complexity semi-algebraic sets in $\mathbb{R}^3$, such that the complexity of the union of any subset of $\mathcal{S}$ of size m is $U(m)$. Then the vertical decomposition of $\mathcal{C}(\mathcal{S})$ can be constructed in $O^*(n^2 + U(n))$ randomized expected time.*

Arrangements in $\mathbb{R}^3$. Let $\mathcal{S}$ be a collection of n semi-algebraic sets (each of constant complexity) in $\mathbb{R}^3$ such that $\mathcal{A}(\mathcal{S})$ has X vertices. The above approach for computing the vertical decomposition of $\mathcal{C}(\mathcal{S})$ can be extended to compute the vertical decomposition of $\mathcal{A}(\mathcal{S})$. The only difference is that we now compute a $(1/(2r))$-cutting of $\mathcal{A}(\mathcal{S})$ of size $O^*(r^2 + r^3X/n^3)$ using Theorem 5.1 (iii). Omitting the straightforward details, we conclude the following result.

THEOREM 5.3. *Let $\mathcal{S}$ be a collection of n constant-complexity semi-algebraic sets in $\mathbb{R}^3$ such that the arrangement $\mathcal{A}(\mathcal{S})$ has X vertices. Then the vertical decomposition of $\mathcal{A}(\mathcal{S})$ can be constructed in $O^*(n^2 + X)$ randomized expected time.*

Lower envelopes in four dimensions. Let $\mathcal{F}$ be a collection of n trivariate semi-algebraic functions of constant complexity. Our goal is to construct the vertical decomposition of E^-, the portion of $\mathcal{A}(\mathcal{F})$ lying below the lower envelope E of $\mathcal{F}$.

We briefly recall how the vertical decomposition is defined. We iterate over the functions of $\mathcal{F}$. For each function $a \in \mathcal{F}$, we form the 2D intersection surfaces $a \cap b$, for $b \in \mathcal{F} \setminus \{a\}$, which we denote for short as ab. We project these surfaces onto the xyz-space, and construct the vertical decomposition of the complement $\mathcal{C}_a$ of the union $\mathcal{U}_a$ as defined in Section 3. As in the basic construction in Section 2, the key step is to find all the vertical visibilities within $\mathcal{C}_a$. Each such visibility is between two edges, each of which is the intersection of two of the surfaces ab (for a fixed). We denote for short the intersection curve of ab and ac as abc. That is, we need to find all the 5-tuples (a, b, c, d, e) of distinct functions of $\mathcal{F}$, such that abc and ade form a vertical visibility (in the z-direction) within $\mathcal{C}_a$. Once we have found all these 5-tuples, completing the representation of the vertical decomposition can be carried out in a routine manner, similar to that used in the three-dimensional case reviewed earlier, which, for this setting, takes overall $O^*(n^3)$ time.

To construct the above visibilities, we proceed as above. Namely, we construct a $(1/(2r))$-cutting Ξ of E^- of size $O^*(r^3)$ using Theorem 5.1 (ii). For each prism $\tau \in \Xi$, let $\mathcal{F}_\tau$ be its conflict list plus the $O(1)$ functions that define τ. We process recursively each prism cell τ, where at the bottom of the recursion we report all pairs of vertical visibilities between the edges of $\mathcal{F}_\tau$ in a brute force manner.

We claim that, for each vertical visibility (in the full collection $\mathcal{F}$) defined by a 5-tuple (a, b, c, d, e), all five functions appear in the conflict list of the same prism $\tau \in \Xi$, so the visibility will be found in the corresponding recursive step (in fact, as just described, it will be found at some leaf of the recursion). Indeed, let ζ be the z-vertical segment in the xyz-space that defines the visibility, with endpoints on abc and on ade. Let ζ^+ be the lifting of ζ to the graph of a. Then ζ^+ is fully contained in E_a, and in fact no function graph crosses the downward vertical curtain erected (in the w-direction) from ζ^+.

We claim that ζ^+ is fully contained in a prism $\tau \in \Xi$, from which the previous claim follows readily. Suppose to the contrary that this is not the case, so ζ^+ crosses the boundary of such a prism. Since ζ^+, or rather ζ, is in the z-direction, it follows that ζ must hit the floor or the ceiling, in the z-direction, of a prism of the three-dimensional decomposition of the minimization diagram, which, by construction, lies on some (xyz-projection of an) intersection surface, say uv. This however is impossible, since no such surface can cross the interior of ζ, which is fully contained in $\mathcal{C}_a$, which is disjoint from all such surface projections. This establishes the correctness of the procedure and yields the following:

THEOREM 5.4. *Let $\mathcal{F}$ be a collection of n trivariate semi-algebraic functions of constant complexity. Then the vertical decomposition of portion of $\mathcal{A}(\mathcal{F})$ lying below the lower envelope of $\mathcal{F}$ can be constructed in randomized expected time $O^*(n^3)$.*

6 Nearest Neighbor Searching amid Lines in $\mathbb{R}^3$

We now turn our attention to nearest-neighbor-searching problems involving points and lines in $\mathbb{R}^3$. In this section, we present a linear-size data structure for preprocessing a set L of n lines in $\mathbb{R}^3$ into a data structure so that for a query point $p \in \mathbb{R}^3$, the line of L nearest to p can be reported quickly (more quickly than what can be obtained by the standard machinery). Using standard techniques (e.g., parametric search) [1, 9], a nearest-neighbor query, referred to as an NN query on L, can be reduced to answering $O^*(1)$ *sphere-intersection-detection* queries on L. That is, we want to preprocess L into a data structure that can efficiently determine whether a query sphere σ intersects any of the lines in L.

Overall data structure. Our overall data structure is based on the following technical property, originally proved by Mohaban and Sharir [42]. Let ℓ be a line in $\mathbb{R}^3$, and let σ_p be a sphere, centered at a point p. Let V_ℓ be the vertical plane that contains ℓ, and let H_ℓ be the plane that contains ℓ and is orthogonal to V_ℓ. We say that ℓ is *lower* (resp., *higher*) than σ_p if p lies above (resp., below) H_ℓ; see Figure 1.

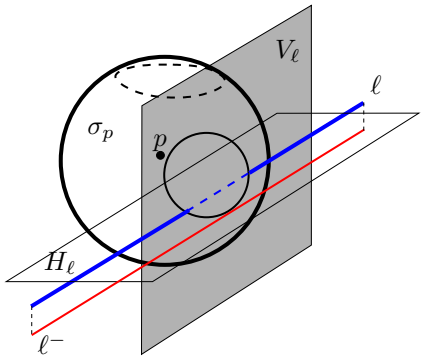

Figure 1: Illustration of condition (ii-) of Lemma 6.1. Here ℓ is lower than σ_p.

LEMMA 6.1. ([42]) *Assuming that ℓ is lower than σ_p (using the above notation), ℓ intersects σ_p if and only if the following two conditions hold:*

(i) *The xy-projections of ℓ and of σ_p intersect, and*

(ii-) *ℓ lies above the parallel line ℓ^- that lies in V_ℓ and is tangent to σ_p from below.*

Symmetrically, assuming that ℓ is higher than σ_p, ℓ intersects σ_p if and only if (i) holds and

(ii+) *ℓ lies below the parallel line ℓ^+ that lies in V_ℓ and is tangent to σ_p from above.*

We describe a linear-size data structure that, for a query sphere σ, determines whether any line of L that is lower than σ intersects σ. (A similar data structure can be constructed for detecting whether any line of L that is higher than σ intersects σ.) We thus need a data structure that, for a query sphere σ, returns YES if a line in ℓ satisfies the following three conditions, as in Lemma 6.1:

(C1) the xy-projections of ℓ and σ intersect,

(C2) ℓ is lower than σ, and

(C3) ℓ lies above the parallel line ℓ^- that lies in V_ℓ and is tangent to σ_p from below.

We use a multi-level partition tree [1, 2] for answering queries of this kind. In particular, we construct a 3-level partition tree, each of whose nodes v stores a "canonical" subset $L_v \subseteq L$. The first-level tree identifies the subset of lines that satisfy condition (C1) for the given query. Since a line in $\mathbb{R}^2$ requires two parameters, (C1) can be formulated as a two-dimensional semi-algebraic range query of a very simple nature—the inequality that we need to test just involves the absolute value of a linear expression. Thus the first level is a 2-dimensional partition tree for semi-algebraic range queries of this simple kind [12, 41]. As shown in [42], and easy to see, (C2) just amounts to testing whether the center of the sphere lies above the respective planes H_ℓ, so it can be formulated as a 3-dimensional halfspace range query. For each node u of the first-level tree, we construct a 3-dimensional partition tree for halfspace range searching, on the subset of lines L_u associated with u, as a second-level tree. Finally, for each node v of every second-level tree, we construct a third-level partition tree on L_v, the subset of lines associated with v, which tests for (C3). We present below a linear-size data structure that can test condition (C3) in $O^*(n^{2/3})$ time (actually, in $O^*(|L_v|^{2/3})$ time). For a query sphere σ, the first two levels of the partition tree return the subset of lines that satisfy conditions (C1) and (C2) as the union of a few canonical subsets (see below for a precise statement). For each of these canonical subsets L_v, the third-level tree constructed on L_v is used to test whether any line in L_v satisfies (C3). If the answer is YES, then we conclude that σ intersects a line of L_v and return YES. Since the query time at each level is $O^*(n^{2/3})$ (it is actually smaller for the first level), the properties of multi-level partition trees (see, e.g., Theorem A.1 in the appendix of [2]), imply that the overall query time is also $O^*(n^{2/3})$. The overall size of the data structure is $O(n)$.[6]

Sphere-intersection query for lines lower than the sphere. Let L be a set of n lines in $\mathbb{R}^3$. We wish to preprocess L into a linear-size data structure that, for a query sphere σ satisfying conditions (C1) and (C2) for all lines in L, can determine in $O^*(n^{2/3})$ time whether σ intersects any line of L. We work in the 4-dimensional parametric space of lines, denoted by $\mathbb{L}$, where a line ℓ is represented by the point $\ell^* = (a, b, c, d)$ and the equations defining ℓ are $y = ax + c$, $z = bx + d$; $\mathbb{L}$ is thus identified[7] with $\mathbb{R}^4$. Put $L^* = \{\ell^* \mid \ell \in L\}$. A sphere σ is associated with a surface (patch) $\gamma_\sigma \subset \mathbb{L}$, which is the locus of points ℓ^* such that the corresponding line ℓ is tangent to σ from below. Let γ_σ^+ be the set of points lying on or above γ_σ in the d-direction; γ_σ^+ is a semi-algebraic set of constant complexity. It is easily seen that a line ℓ satisfying conditions (C1) and (C2) intersects σ if and only if ℓ^* lies in γ_σ^+. Let Γ be the collection of all sets γ_σ^+ such that σ satisfies (C1) and (C2) for all lines in L. Thus the sphere-intersection query for a sphere σ in our setting reduces to semi-algebraic range-emptiness query in L^* with $\gamma_\sigma^+ \in \Gamma$. Using the known and standard partition tree mechanism [12, 41], this query can be answered in $O^*(n^{3/4})$ time, but we show how to improve the query time to $O^*(n^{2/3})$.

We follow the approach of Matoušek [40] and of Sharir and Shaul [46] for answering the range-emptiness query. We need a couple of definitions. Let $P \subset \mathbb{L}$ be a set of n points. For a parameter $k \geq 0$, we call a semi-algebraic set $\gamma \subset \mathbb{L}$, which semi-unbounded in the negative d-direction, *k-shallow* if $|P \cap \gamma| \leq k$. For a parameter $r \geq 1$, we call a family $\Pi = \{(P_1, \Delta_1), \ldots, (P_u, \Delta_u)\}$ a $(1/r)$-partition for P if (i) $\{P_1, \ldots, P_u\}$ is a partition of P, (ii) $n/2r \leq |P_i| \leq n/r$, and (iii) $P_i \subset \Delta_i$ where $\Delta_i \subseteq \mathbb{L}$ is a semi-algebraic set of constant complexity, referred to as a cell of Π. The *crossing number* of Π for a semi-algebraic set τ, denoted by $\chi(\Pi, \tau)$, is the number of cells of Π intersected by the boundary of τ. The crossing number of Π for a family Ξ of semi-algebraic sets, denoted by $\chi(\Pi, \Xi)$, is defined as $\max_{\tau \in \Xi} \chi(\Pi, \tau)$.

A major ingredient of the approach in [40, 46] is to construct a so-called *test set* Φ of a small number of semi-algebraic sets, which represent well all query semi-algebraic sets that are shallow. The following lemma of Sharir and Shaul [46, Theorem 3.2] summarizes the key property:

[6] A straightforward application of the multi-level data-structure framework leads to a data structure of size $O^*(n)$. But, using well known machinery, the size can be improved to $O(n)$ while keeping the query time $O^*(n^{2/3})$ by constructing secondary structures only at some of the nodes.

[7] For convenience (and with no loss of generality if one assumes general position), we ignore the fact that this space is actually projective.

LEMMA 6.2. ([46]) *Let P be a set of n points in $\mathbb{R}^d$, for some $d \geq 1$, and let Γ be a (possibly infinite) family of semi-algebraic sets of constant complexity. Let $r \geq 1$ be a parameter, and let Φ be another finite collection (not necessarily a subset of Γ) of semi-algebraic sets of constant complexity with the following properties:*

(i) *Every set in Φ is (n/r)-shallow with respect to P.*

(ii) *The complement of the union of any m sets of Φ can be decomposed into at most $\zeta(m)$ "elementary cells" (semi-algebraic sets of constant complexity) for any $m \geq 1$, where $\zeta(m)$ is a suitable monotone increasing superlinear function of m.*

(iii) *Any (n/r)-shallow set $\gamma \in \Gamma$ can be covered by the union of at most δ ranges of Φ, where δ is a constant (independent of r).*

Then there exists a $(1/r)$-partition Π of P such that for any (n/r)-shallow range $\gamma \in \Gamma$, $\chi(\Pi, \gamma) = O(r/\zeta^{-1}(r) + \log r \log |\Phi|)$ if $\zeta(r)/r^{1+\varepsilon}$ is monotonically increasing for some (arbitrarily small) constant $\varepsilon > 0$, and $\chi(\Pi, \gamma) = O(r \log r/\zeta^{-1}(r) + \log r \log |\Phi|)$ otherwise. Furthermore, Π can be constructed in $(|\Phi| + n)r^{O(d)}$ expected time assuming Φ is given.

As shown in [40, 46], using Lemma 6.2 and assuming that $|\Phi| = r^{O(d)}$, one can construct a partition tree of linear-size that can determine in $O^*(n/\zeta^{-1}(n))$ time whether $\gamma \cap P \neq \emptyset$, for any query range $\gamma \in \Gamma$. We present an algorithm below for constructing a test set Φ of size $r^{O(1)}$ for our setup so that $\zeta(m) = O^*(m^3)$ and $\delta = 1$, which in turn yields a linear-size data structure for sphere intersection queries with $O^*(n^{2/3})$ query time, as desired.

Constructing a test set. To construct the test set, we also use the 4-dimensional parametric space $\mathbb{S}$ of spheres in $\mathbb{R}^3$, where a sphere σ of radius r centered at a point p is mapped to the point $\sigma^* = (p, r) \in \mathbb{S}$; $\mathbb{S}$ can thus be identified with $\mathbb{R}^4$. A line ℓ in $\mathbb{R}^3$ is mapped to a surface ω_ℓ, consisting of all points $\sigma^* \in \mathbb{S}$ that represent spheres that touch ℓ from above. As is easily verified, these surfaces are monotone over the xyz-subspace, so that a point σ^* lies above the surface ω_ℓ if and only if ℓ intersects σ, assuming σ and ℓ satisfy (C1) and (C2).[8]

Let $\Omega = \{\omega_\ell \mid \ell \in L\}$ denote the collection of these surfaces. We take a random subset $R \subseteq \Omega$ of $s = cr \log r$ surfaces, for some sufficiently large constant r, and construct the vertical decomposition $\mathrm{VD}(R)$ of the arrangement $\mathcal{A}(R)$; $\mathrm{VD}(R)$ has $O^*(r^4)$ cells [36]. By a standard random-sampling argument [35], each cell of $\mathrm{VD}(R)$ is crossed by at most n/r surfaces of Ω with probability at least $1/2$. If this is not the case, we discard R and choose another random subset, until we find one with the desired property. We choose a subset Ξ of $\mathrm{VD}(R)$, namely, those cells that have at most n/r surfaces of Ω passing *fully below* them. By construction, these cells cover the lowest n/r levels of $\mathcal{A}(\Omega)$, and are contained in the at most $2n/r$ lower levels of $\mathcal{A}(\Omega)$.

Let τ be a cell of Ξ. We now switch to the parametric line-space $\mathbb{L}$, where each point $\sigma^* \in \tau$ becomes the surface γ_σ. We construct the lower envelope of the (infinitely many) surfaces γ_σ over all $\sigma^* \in \tau$. Let $\Phi_\tau \subset \mathbb{L}$ be the set of points lying above the lower envelope. Since τ has constant complexity, Φ_τ is a semi-algebraic surface of constant complexity. A point $\ell^* \in L^*$ lies in Φ_τ if and only if there is a surface γ_σ, with $\sigma^* \in \tau$, that passes below ℓ^*. This happens when, back in $\mathbb{S}$, the surface ω_ℓ (corresponding to the line ℓ) crosses τ or lies below τ. By construction, there are at most $n/r + n/r = 2n/r$ such surfaces. Consequently, Φ_τ is $(2n/r)$-shallow with respect to the points of L^*.

Set $\Phi = \{\Phi_\tau \mid \tau \in \Xi\}$. Φ is a family of $O^*(r^4)$ constant-complexity semi-algebraic surfaces[9] in $\mathbb{L}$, each of which is $(2n/r)$-shallow with respect to L^*. This is our desired test set, as stated in the following lemma. The proof of the lemma is an immediate consequence of our construction.

LEMMA 6.3. *Let σ be a sphere that satisfies (C1) and (C2) with respect to the lines of L and that is (n/r)-shallow with respect to L. Then there exists a semi-algebraic set of Φ that contains σ^*.*

Plugging Lemma 6.3 into Lemma 6.2, Φ is a test set for L^* with respect to the semi-algebraic ranges in Γ, with $\delta = 1$ and $\zeta(m) = O^*(m^3)$. The bound on $\zeta(m)$ follows from Theorem 3.1. Putting everything together, we thus obtain:

THEOREM 6.1. *A set L of n lines in $\mathbb{R}^3$ can be preprocessed, in $O^*(n)$ expected time, into a data structure of size $O(n)$ so that for any query point $p \in \mathbb{R}^3$, the line of L nearest to p can be computed in $O^*(n^{2/3})$ time.*

[8] Informally, this is why we have to distinguish between lines that pass below the sphere and lines that pass above.

[9] By construction, as in [46], these semi-algebraic sets do not correspond to spheres any more, but they are nevertheless semi-algebraic sets of constant complexity.

7 Nearest-Neighbor Queries with Lines in $\mathbb{R}^3$

In this section we consider the converse situation, where queries are lines in $\mathbb{R}^3$. We first consider in Section 7.1 a simpler, yet challenging, case where the input is a set of points in $\mathbb{R}^3$, and then, in Section 7.2, consider the case where the input is a set of lines in $\mathbb{R}^3$. We are interested in a data structure that answers NN queries in $O^*(1)$ time using as little storage as possible.

7.1 Nearest-point queries with lines in $\mathbb{R}^3$ Let P be a set of n points in $\mathbb{R}^3$. Since we are aiming for an $O^*(1)$ query time, we work in the 4-dimensional parametric space $\mathbb{L}$ of (query) lines (the same parametric space used in the previous section), where a line ℓ in $\mathbb{R}^3$, given by the equations $y = ax + c$ and $z = bx + d$, is represented as the point $\ell^* = (a, b, c, d) \in \mathbb{L}$. We begin by describing the distance function between a point and a line in $\mathbb{R}^3$ and the Voronoi diagram that the points of P induce in $\mathbb{L}$.

Distance function, lower envelope, Voronoi diagram. Let $\ell^* = (a, b, c, d) \in \mathbb{L}$. For a fixed pair $a, b \in \mathbb{R}$, the (unnormalized) direction of ℓ, $(1, a, b)$, is fixed. Let H be the plane that is orthogonal to ℓ (i.e., with normal direction $(1, a, b)$) and passes through the origin. Redefine the representation of ℓ so that (c, d) is actually the intersection of ℓ with H, in a suitable canonical coordinate frame within H (we omit here the easy details of specifying this frame, noting that it does depend on (a, b)). Write $u = (1, a, b)$.

For a point $p \in P$, let $p^\downarrow$ denote its projection onto H. Concretely, write $p^\downarrow = p + tu$. The condition for $p^\downarrow$ to lie in H is that $p + tu$ be orthogonal to u (recall that H passes through the origin). That is, we require that

$$(p + tu) \cdot u = p \cdot u + t|u|^2 = 0, \quad \text{or} \quad t = -\frac{p \cdot u}{|u|^2}.$$

That is, we have

$$p^\downarrow = p - \frac{p \cdot u}{|u|^2} u.$$

Write $p^\downarrow = (x_p(a, b), y_p(a, b))$; clearly, these coordinates depend on (a, b). The distance between p and ℓ, denoted by $\mathrm{dist}(p, \ell)$, is then the distance between $p^\downarrow$ and (c, d). That is,

$$\begin{aligned} \mathrm{dist}^2(p, \ell) &= (x_p(a, b) - c)^2 + (y_p(a, b) - d)^2 \\ &= (x_p^2(a, b) + y_p^2(a, b)) - 2cx_p(a, b) - 2dy_p(a, b) + (c^2 + d^2). \end{aligned} \tag{7.1}$$

For a query line ℓ, our goal is to compute $\arg\min_{p \in P} \mathrm{dist}^2(p, \ell)$, the point $p \in P$ that is closest to ℓ, i.e., minimizes (7.1). Since $c^2 + d^2$ is common to all points p, we can drop it, and seek the point p that minimizes

$$f_p(a, b, c, d) = g_p(a, b) - 2cx_p(a, b) - 2dy_p(a, b), \tag{7.2}$$

where $g_p(a, b) = x_p^2(a, b) + y_p^2(a, b)$. Let $\mathcal{F} = \{f_p \mid p \in P\}$ be the resulting set of n 4-variate functions. Consider the lower envelope $E : \mathbb{L} \to \mathbb{R}$ of $\mathcal{F}$ defined as

$$E(a, b, c, d) = \min_{p \in P} f_p(a, b, c, d).$$

The projection of the graph of E onto $\mathbb{L}$, denoted by $\mathsf{M} := \mathsf{M}(P)$, is called the minimization diagram of $\mathcal{F}$. M induces a partition of $\mathbb{L}$, to which we refer as the Voronoi diagram of P in $\mathbb{L}$. Each cell τ of M is associated with a point $p \in P$ that is the nearest neighbor of all lines whose dual points lie in the cell τ. For a query line ℓ, we wish to locate the cell of M containing $\ell^* = (a, b, c, d)$. However, currently we do not know how to preprocess four-dimensional minimization diagrams, like M, into a data structure of size $O^*(n^4)$ for answering point-location queries in $O^*(1)$ time. We manage to address this problem by exploiting the additional structure of the Voronoi cells of M.

Structure of Voronoi cells. For each point $p \in P$, let M_p denote the region of $\mathbb{L}$ where f_p attains E, i.e., the set of cells of M that are associated with p. Let E_p denote the graph of E restricted to M_p, which is a suitable subset of the graph of f_p.

For each $q \in P$, $q \neq p$, let $\sigma_{p,q}$ denote the intersection surface of f_p and f_q, which is a three-dimensional surface that is disjoint from the relative interior of E_p, and does not pass below any point on E_p. It is defined by the equation

$$g_p(a, b) - 2cx_p(a, b) - 2dy_p(a, b) = g_q(a, b) - 2cx_q(a, b) - 2dy_q(a, b).$$

Assuming $y_q(a,b) \neq y_p(a,b)$, we define a trivariate function $\psi_{p,q} : \mathbb{L}^{(d)} \to \mathbb{R}$, where $\mathbb{L}^{(d)} \subset \mathbb{L}$ is the 3-dimensional hyperplane $d = 0$, as follows:

$$d = \psi_{p,q}(a,b,c) := \frac{g_q(a,b) - g_p(a,b)}{2(y_q(a,b) - y_p(a,b))} - \frac{x_q(a,b) - x_p(a,b)}{y_q(a,b) - y_p(a,b)} c. \tag{7.3}$$

The surface $\sigma_{p,q}$ partitions $\mathbb{L}$ into the two regions

$$K_{p,q}^{(d+)} = \{\ell^* \in \mathbb{L} \mid f_p(\ell^*) \leq f_q(\ell^*)\} \quad \text{and} \quad K_{p,q}^{(d-)} = \{\ell^* \in \mathbb{L} \mid f_p(\ell^*) \geq f_q(\ell^*)\}.$$

Then $\mathsf{M}_p = \bigcap_{q \in P \setminus \{p\}} K_{p,q}^{+}$. By (7.3), we can write $K_{p,q}^{(d+)}$ as

$$K_{p,q}^{(d+)} = \{(a,b,c,d) \in \mathbb{L} \mid y_q(a,b) - y_p(a,b) \geq 0,\ d \leq \psi_{p,q}(a,b,c)\} \bigcup \\ \{(a,b,c,d) \in \mathbb{L} \mid y_q(a,b) - y_p(a,b) \leq 0,\ d \geq \psi_{p,q}(a,b,c)\}.$$

To simplify this representation, we define two functions $\psi_{p,q}^{+}, \psi_{p,q}^{-} : \mathbb{L}^{(d)} \to \mathbb{R}$ by:

$$\psi_{p,q}^{(d+)}(a,b,c) = \begin{cases} \psi_{p,q}(a,b,c) & y_q(a,b) - y_p(a,b) \geq 0 \\ +\infty & \text{otherwise} \end{cases} \\ \psi_{p,q}^{(d-)}(a,b,c) = \begin{cases} \psi_{p,q}(a,b,c) & y_q(a,b) - y_p(a,b) \leq 0 \\ -\infty & \text{otherwise.} \end{cases} \tag{7.4}$$

Then we can write

$$K_{p,q}^{(d+)} = \{(a,b,c,d) \in \mathbb{L} \mid \psi_{p,q}^{(d-)}(a,b,c) \leq d \leq \psi_{p,q}^{(d+)}(a,b,c)\}. \tag{7.5}$$

In other words, M_p is the *sandwich region* between the lower envelope (with respect to the d-direction) $E_p^{(d-)}$ of the functions $\psi_{p,q}^{(d+)}$ and the upper envelope $E_p^{(d+)}$ of the functions $\psi_{p,q}^{(d-)}$, for $q \in P \setminus \{p\}$. See Figure 2 for an illustration.

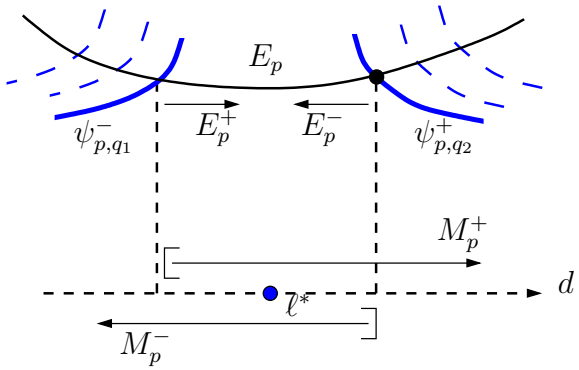

Figure 2: The structure of the decomposition of the lower envelope and the minimization diagram of the sample. To simplify the figure, the superscripts $(d+)$ and $(d-)$ have been suppressed.

We can thus write M_p as $\mathsf{M}_p^{(d-)} \cap \mathsf{M}_p^{(d+)}$, where $\mathsf{M}_p^{(d-)}$ (resp., $\mathsf{M}_p^{(d+)}$) is the region below the lower envelope $E_p^{(d-)}$ (resp., above the upper envelope $E_p^{(d+)}$) in the d-direction.

Note that the above construction is symmetric in c and d, as each function f_p is linear in both c and d. We can therefore repeat the whole construction, switching between c and d. The analysis is fully symmetric, with obvious modifications, such as having $x_q(a,b) - x_p(a,b)$ in the denominators in (7.2), and similar straightforward changes. M_p can now be written as $\mathsf{M}_p^{(c-)} \cap \mathsf{M}_p^{(c+)}$, where $\mathsf{M}_p^{(c-)}$ (resp., $\mathsf{M}_p^{(c+)}$) is the region below (resp., above),

in the c-direction, the lower envelope $E_p^{(c-)}$ (resp., upper envelope $E_p^{(c+)}$) of the corresponding set of trivariate functions $\psi_{p,q}^{(c-)}$ (resp., $\psi_{p,q}^{(c+)}$) defined analogously to $\psi_{p,q}^{(d-)}$ (resp., $\psi_{p,q}^{(d+)}$).

We conclude this discussion with the following observation, which will be the key to the performance of our data structure and the query procedure.

LEMMA 7.1. *Let $\ell^* = (\ell_a, \ell_b, \ell_c, \ell_d) \in \mathbb{L}$, and let p be a point of P. Let $\rho^{(d)}$ (resp., $\rho^{(c)}$) denote the line in the d-direction (resp., c-direction) in $\mathbb{L}$ passing through ℓ^*, and let $\gamma^{(d)}$ (resp., $\gamma^{(c)}$) denote the curve on (the graph of) f_p traced over the line $\rho^{(d)}$ (resp., $\rho^{(c)}$). Let q be a point of P that is nearer to ℓ than p, assuming that such a point exists, i.e., $f_q(\ell^*) < f_p(\ell^*)$. Then f_q intersects either $\gamma^{(d)}$ or $\gamma^{(c)}$. Furthermore if f_q intersects $\gamma^{(d)}$ at a point $w = (w_a, w_b, w_c, w_d)$ such that $w_d > \ell_d$ (resp., $w_d < \ell_d$) then we have $w_d = \psi_{p,q}^{(d-)}(w_a, w_b, w_c)$ (resp., $w_d = \psi_{p,q}^{(d+)}(w_a, w_b, w_c)$). A similar property holds if f_q intersects $\gamma^{(c)}$.*

Proof. Suppose f_q does not intersect $\gamma^{(d)}$. Then we would have, using (7.2),

$$g_q(a,b) - 2cx_q(a,b) - 2dy_q(a,b) < g_p(a,b) - 2cx_p(a,b) - 2dy_p(a,b)$$

for every d. Since a, b, c are fixed along $\gamma^{(d)}$, this can happen only when $y_q(a,b) = y_p(a,b)$. Repeating the same argument for $\gamma^{(c)}$, if f_q does not intersect $\gamma^{(c)}$, then $x_q(a,b) = x_p(a,b)$. Therefore, if f_q does not intersect either of these curves then we also have, by definition, $g_q(a,b) = g_p(a,b)$, which implies that $f_q(\ell^*) = f_p(\ell^*)$, i.e., p and q are equidistant from ℓ. This contradicts the assumption that q is (strictly) nearer to ℓ than p.

Thus f_q intersects one of the curves, say, for specificity, that it intersects $\gamma^{(d)}$. Again, by (7.2), f_q intersects $\gamma^{(d)}$ at a unique point $w = (w_a, w_b, w_c, w_d)$, with $w_d = \psi_{p,q}(w_a, w_b, w_c)$. If $w_d > \ell_d$ (resp., $w_d < \ell_d$), then by (7.4), we must have $w_d = \psi_{p,q}^{(d-)}(w_a, w_b, w_c)$ (resp., $w_d = \psi^{(d+)}(w_a, w_b, w_c)$). This completes the proof of the lemma. □

We are now ready to describe the data structure based on the above lemma.

Overall data structure. Fix some sufficiently large constant parameter $r > 0$. We choose a random subset $R \subseteq P$ of $cr \log r$ points, for a suitable absolute constant $c > 0$. We construct the Voronoi diagram $\mathsf{M}(R)$ of R. For every point $p \in R$, we construct $\mathsf{M}_p^{(d-)}, \mathsf{M}_p^{(d+)}, \mathsf{M}_p^{(c-)}, \mathsf{M}_p^{(c+)}$, as defined above (with respect to $\mathsf{M}(R)$). Let $\Xi_p^{(d-)} = \mathrm{VD}(\mathsf{M}_p^{(d-)})$ be the vertical decomposition of $\mathsf{M}_p^{(d-)}$. Similarly define, $\Xi_p^{(d+)}, \Xi_p^{(c-)}, \Xi_p^{(c+)}$. Let Ξ be the set of cells in all these $4|R|$ vertical decompositions. By Theorem 3.1, $|\Xi| = O^*(r \cdot r^3) = O^*(r^4)$, and by Theorem 5.4, Ξ can be constructed in a total of $O^*(r^4)$ expected time.

We define a *conflict list* L_τ for every $\tau \in \Xi$, as follows. For each point $p \in R$ and each cell τ of $\Xi_p^{(d-)}$ (resp., $\Xi_p^{(d+)}$), we define $P_\tau \subset P$ to be the subset of points $q \in P$ for which the surface $d = \psi_{p,q}^{(d+)}$ (resp., $d = \psi_{p,q}^{(d^-)}$) crosses τ. With a suitable choice of c, the size of each conflict list is at most n/r, with high probability, because, by construction, for a cell τ of $\mathsf{M}_p^{(d-)}$ (resp., $\mathsf{M}_p^{(d+)}$), none of the surfaces $d = \psi_{p,u}^+$ (resp., $d = \psi_{p,u}^-$), for $u \in R \setminus \{p\}$, intersect τ [35]. Similarly we define the conflict lists of cells in $\Xi_p^{(c-)}, \Xi_p^{(c+)}$; their sizes are also all at most n/r, with high probability.

For each cell $\tau \in \Xi$, we recursively build the data structure on P_τ. The recursion stops when the size of a subproblem becomes smaller than some fixed absolute constant n_0. Since there are $O^*(r^4)$ subproblems of size at most n/r each, a straightforward analysis shows that the size of the overall structure is $O^*(n^4)$, and that it can be constructed in $O^*(n^4)$ expected time.

Query procedure. A query with a line ℓ is processed as follows. We compute the nearest neighbor of ℓ in R, which we call p. Next, we compute the cells $\tau^{(d-)}, \tau^{(d+)}, \tau^{(c-)}, \tau^{(c+)}$ of $\mathsf{M}_p^{(d-)}, \mathsf{M}_p^{(d+)}, \mathsf{M}_p^{(c-)}, \mathsf{M}_p^{c+)}$, respectively, that contain ℓ^*. All this is done in brute force and takes $O^*(1)$ time. If P contains a point q that is nearer to ℓ than p, then by Lemma 7.1, f_q intersects either the curve $\gamma^{(d)}$ or $\gamma^{(c)}$. Suppose f_q intersects $\gamma^{(d)}$ at a point $w = (w_a, w_b, w_c, w_d)$. Again, by Lemma 7.1, if $w_d \geq \ell_d$, then $w_d = \psi_{p,q}^-(w_a, w_b, w_c)$, implying that $w \in \tau^{d+}$ and thus q belongs to the conflict list $P_{\tau^{d+}}$. Similarly, if $w_d < \ell_d$, then q belongs to the conflict list $P_{\tau^{d-}}$. A symmetric analysis applies when f_q intersects $\gamma^{(c)}$. In summary, if q is closer to ℓ than p then q lies in the conflict lists of one of $\tau^{(d-)}, \tau^{(d+)}, \tau^{(c-)}, \tau^{(c+)}$. Hence, we need to search recursively in these four subproblems, and return the nearest point among p and the points returned by these four recursive subproblems.

Since we recurse in four subproblems, each of size at most n/r (and r can be chosen to be a sufficiently large constant), the total query time is $O^*(1)$ (it is not polylogarithmic, though). We thus obtain the following result:

THEOREM 7.1. *A given set P of n points in $\mathbb{R}^3$ can be preprocessed, in $O^*(n^4)$ expected time, into a data structure of size $O^*(n^4)$, so that, for any query line $\ell \in \mathbb{R}^3$, the point of P nearest to ℓ can be computed in $O^*(1)$ time.*

7.2 Nearest-line queries with lines in $\mathbb{R}^3$ Next, we show that the machinery in the preceding subsection can be extended (with a couple of twists—see below) to obtain a line NN-searching data structure, with the same asymptotics performance, when the input is a set L of n lines in $\mathbb{R}^3$, and we want to find the line nearest to a query line. We first describe the two new challenges we face in dealing with lines as input, and explain how to address them, and then describe the overall data structure.

We use the same representation (a, b, c, d) for the query line ℓ, using the orthogonal plane H as before. Thus ℓ is represented as the same point $\ell^* \in \mathbb{L}$. For a line $\lambda \in L$, let $\lambda^{\downarrow}$ denote the projection of λ onto H. A crucial observation, which is easy to verify, is that

$$f_\lambda(\ell^*) := \operatorname{dist}(\ell, \lambda) = \operatorname{dist}(\ell, \lambda^{\downarrow}) = \operatorname{dist}((c, d), \lambda^{\downarrow}).$$

The equation of $\lambda^{\downarrow}$, in the canonical coordinate frame within H, is of the form

$$\xi_\lambda(a, b)x + \eta_\lambda(a, b)y + \zeta_\lambda(a, b) = 0,$$

where we normalize the coefficients so that $\xi_\lambda^2(a, b) + \eta_\lambda^2(a, b) = 1$. Hence,

$$f_\lambda(a, b, c, d) = \left|\xi_\lambda(a, b)c + \eta_\lambda(a, b)d + \zeta_\lambda(a, b)\right|. \tag{7.6}$$

Except for the absolute value, (7.6) is linear in c and d, as in the preceding analysis, a property that has been crucial for the analysis there, and will be crucial for the analysis here too.

We handle the absolute value as follows. Orient each line $\lambda \in L$ in an arbitrary (but fixed) manner, say in the positive x-direction, and similarly orient each query line ℓ. If we know the relative orientation of ℓ and λ, then we also know the sign in the expression for $f_\lambda(\ell^*)$. In fact, we can reduce the setup in such a way that allows us to assume that the sign is positive if and only if the relative orientation is positive. For a line $\lambda \in L$, we define the surface $\sigma_\lambda \subset \mathbb{L}$, which is the locus of all points $\ell^* \in \mathbb{L}$ such that ℓ touches λ. It partitions $\mathbb{L}$ space into two portions, one consisting of points representing lines that are positively oriented with respect to λ, and the other consists of points with negative orientations. We construct a data structure on these surfaces that, for a query (oriented) line ℓ, partitions the set of all lines of L into $O(\log n)$ "canonical" subsets such that, for every canonical subset, either all its lines are positively oriented with respect to ℓ or all of them are negatively oriented.

In view of the above discussion, let us assume that the query line has positive orientation with respect to all lines in L, and that this corresponds to a positive sign of the expression in (7.6). We construct a data structure on L using, more or less, the same machinery as in Section 7.1, exploiting the double linearity (in c and d) of the distance functions. Here we face the second challenge. Recall that we basically showed in Lemma 7.1 that if f_p and f_q do not cross along the lines ρ_d, ρ_c, then we have $x_p(a, b) = x_q(a, b)$ and $y_p(a, b) = y_q(a, b)$, and thus the free terms $g_p(a, b)$ and $g_q(a, b)$ are also equal, implying that p and q are equidistant from the query line ℓ. Here, in contrast, if $f_\lambda, f_{\lambda'}$, for two distinct lines $\lambda, \lambda' \in L$, do not cross along ρ_c, ρ_d, we can show, using the same reasoning as before, but based on (7.6), that $\xi_\lambda(a, b) = \xi_{\lambda'}(a, b)$ and $\eta_\lambda(a, b) = \eta_{\lambda'}(a, b)$ (actually, one equality suffices, because of our normalization). However, now it no longer follows that $\zeta_\lambda(a, b) = \zeta_{\lambda'}(a, b)$. That is, the projected lines (on $H(a, b)$) could be parallel, and λ' could still be (strictly) nearer to ℓ than λ.

To address this issue we proceed as follows. For each pair of lines λ, λ' in L, let $\beta_{\lambda,\lambda'}$ denote the one-dimensional locus of all (a, b) for which the projections of λ and λ' onto H are parallel; this is the curve $\xi_\lambda(a, b) = \xi_{\lambda'}(a, b)$. For each λ in the sample R, we construct the two-dimensional arrangement $\mathcal{A}_\lambda$ of the curves in $\{\beta_{\lambda,\lambda'} \mid \lambda' \in L \setminus \{\lambda\}\}$, in the (a, b)-plane. For a query dual point $\ell^* = (\ell_a, \ell_b, \ell_c, \ell_d)$, we locate the point (ℓ_a, ℓ_b) in $\mathcal{A}_\lambda$ and find the set L_{par} of the curves $\beta_{\lambda,\lambda'}$ that contain the point (ℓ_a, ℓ_b) to determine the lines of L whose projections onto $H(a, b)$ are parallel to λ. (See below how the algorithm handles sets L_{par} of large size.)

We now describe the overall data structure and the query procedure by incorporating these observations in the data structure described in Section 7.1.

Overall data structure. We build a three-level data structure. Let $\Sigma = \{\sigma_\lambda \mid \lambda \in L\}$. At the top-level, we construct a tree data structure $\mathcal{T}^{(1)}$ for answering point-enclosure queries on Σ, using the algorithm in [3]. Each node u of $\mathcal{T}^{(1)}$ is associated with a *canonical subset* $L_u \subseteq L$ of lines. For a query line ℓ, querying with ℓ^* in $\mathcal{T}^{(1)}$ partitions the lines of L into $O(\log n)$ canonical subsets, each associated with one of its nodes, such that all lines in one subset are either positively oriented with respect to ℓ or all of them are negatively oriented.

For each node v of $\mathcal{T}^{(1)}$, we construct two second-level data structures $\mathcal{T}_v^{(2+)}, \mathcal{T}_v^{(2-)}$ on the canonical subset L_v—one assuming that the sign in (7.6) is positive and the other assuming that it is negative. These structures are

constructed by following and adapting the construction in Section 7.1, using the expressions in (7.6) (without the absolute value) instead of those in (7.2), following both the c- and d-directions, and using partial lower envelopes within the minimization diagram. Each of $\mathcal{T}_v^{(2+)}, \mathcal{T}_v^{(2-)}$ essentially consists of several tree data structures. Each node w of $\mathcal{T}^{(2+)}$ or $\mathcal{T}^{(2-)}$ is also associated with a subset $L_w \subseteq L_v$ of lines. We choose a random subset $R_w \subset L_w$ of size $cr \log r$, for some constant $c \geq 1$, and construct, as in Section 7.1, a total of $O^*(r^4)$ subproblems, each of size at most $|L_w|/r$. In addition, we now store the following third-level structure at w: For each line $\lambda \in R$, we construct the two-dimensional arrangement $\mathcal{A}_\lambda$ of the curves $\mathcal{B}_\lambda = \{\beta_{\lambda,\lambda'} \mid \lambda' \in L_w \setminus R_w\}$ and preprocess it for point-location queries. If the input lines are in general position, then at most two curves of $\mathcal{B}_\lambda$ pass through any point (a, b), and we simply store them. Otherwise, many curves of $\mathcal{B}_\lambda$ may pass through a vertex $\chi = (\chi_a, \chi_b)$ of $\mathcal{A}_\lambda$. Let $L_\chi \subseteq L_w \setminus R_w$ be the subset of lines whose curves are incident on χ. We store L_χ in a sorted order (by the ordering of their projections on the plane $H(\chi_a, \chi_b)$) so that for a query line ℓ of the form $\ell^* = (\chi_a, \chi_b, \ell_c, \ell_d)$, we can find the line in L_χ nearest to ℓ in $O(\log n)$ time. The total size of this third-level data structure over all lines of R is $|R| \cdot O(|L_w|^2) = O(|L_w|^2)$. Using the properties of multi-level data structures, one can show that the overall size of the data structure is $O^*(n^4)$ and that it can be constructed in $O^*(n^4)$ expected time.

Query procedure. For a query line ℓ, we first search in $\mathcal{T}^{(1)}$ with ℓ^* and compute a partition of L into $O(\log n)$ canonical subsets, each associated with a node of $\mathcal{T}^{(1)}$, such that each subset is positively oriented or negatively oriented with respect to ℓ. For each such node v, if the lines in L_v have positive (resp., negative) orientation with respect to ℓ, we search in $\mathcal{T}_v^{(2+)}$ (resp. $\mathcal{T}_v^{(2-)}$) with ℓ^*, as in Section 7.1. At each second-level node w visited by the query procedure, if λ is the nearest neighbor of ℓ in R_w, we recursively search in the four corresponding children of v as in the previous section. In addition, we locate the point (ℓ_a^*, ℓ_b^*) in the arrangement $\mathcal{A}_\lambda$ to find, in $O(\log n)$ time, the nearest neighbor of ℓ among the line of $L_w \setminus R$ whose projections on $H(\ell_a, \ell_b)$ are parallel to that of λ, if any such lines exist. Following the same analysis as above, the overall query time remains $O^*(1)$. Putting everything together, we obtain the following result:

THEOREM 7.2. *A given set L of n lines in $\mathbb{R}^3$ can be preprocessed, in $O^*(n^4)$ expected time, into a data structure of size $O^*(n^4)$, so that, for any query line $\ell \in \mathbb{R}^3$, the line of L nearest to ℓ can be computed in $O^*(1)$ time.*

8 Conclusion

In this paper, we settled in the affirmative a few long-standing open problems involving the vertical decomposition of various substructures of arrangements in $d = 3, 4$ dimensions. In particular, we obtained sharp bounds on the vertical decomposition of the complement of the union of a family of semi-algebraic sets in $\mathbb{R}^3$ of constant complexity, and of the lower envelope of a family of semi-algebraic trivariate functions of constant complexity. We also obtained an output-sensitive bound on the size of the vertical decomposition of the full arrangement of a family of semi-algebraic sets in $\mathbb{R}^3$ of constant complexity. These results lead to efficient algorithms for constructing the vertical decompositions themselves, for constructing $(1/r)$-cuttings of the above substructures of arrangements, and for answering point-enclosure queries. Finally, we applied these results to obtain faster data structures for various basic proximity problems involving lines and points in $\mathbb{R}^3$.

We conclude by mentioning a few open problems:

- The major open question is, of course, to improve the complexity of the vertical decomposition of the arrangement of a family of semi-algebraic sets in $\mathbb{R}^d$ for $d \geq 5$. But an immediate open question is whether the techniques developed in this paper can be extended to obtain improved bounds on the vertical decomposition of various substructures of arrangements (besides lower or upper envelopes) in $\mathbb{R}^4$.
- No non-trivial lower bounds are known for nearest-neighbor data structures involving lines in $\mathbb{R}^3$. This raises the question whether the data structures presented in Sections 6 and 7 are (almost) best possible, or whether one can obtain significantly faster data structures. For example, can the nearest neighbor of a line amid a set of points in $\mathbb{R}^3$ be returned in $O(\log n)$ time using an $O^*(n^3)$ size data structure?

References

[1] P. K. Agarwal, Simplex range searching and its variants: A review, in *Journey through Discrete Mathematics: A Tribute to Jiří Matoušek*, M. Loebl, J. Nešetřil, and R. Thomas (editors), Springer Verlag, Berlin-Heidelberg, 2017, pp. 1–30.

[2] P. K. Agarwal, B. Aronov, E. Ezra, M. J. Katz, and M. Sharir, Intersection queries for flat semi-algebraic objects in three dimensions and related problems. https://doi.org/10.48550/arXiv.2203.10241. (A preliminary version appeared in *Proc. 38th Intl. Sympos. Comput. Geom.*, pages 4:1–4:14, 2022.)
[3] P. K. Agarwal, B. Aronov, E. Ezra, and J. Zahl, An efficient algorithm for generalized polynomial partitioning and its applications, *SIAM J. Comput.* 50 (2021), 760–787.
[4] P. K. Agarwal, B. Aronov, and M. Sharir, Computing envelopes in four dimensions with applications, *SIAM J. Comput.* 26(6) (1997), 1714–1732.
[5] P. K. Agarwal, A. Efrat, and M. Sharir, Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications, *SIAM J. Comput.* 29(3) (1999), 912–953.
[6] P. K. Agarwal and J. Erickson, Geometric range searching and its relatives, in *Advances in Discrete and Computational Geometry*, volume 223 of *Contemp. Math.*, pages 1–56, AMS Press, Providence, RI, 1999.
[7] P. K. Agarwal and E. Ezra, Line intersection searching amid unit balls in 3-Space, *Proc. 39th Intl. Sympos. Comput. Geom.*, pages 5:1–5:14, 2023.
[8] P. K. Agarwal, E. Ezra, and M. Sharir, Vertical decomposition in 3D and 4D with applications to line nearest-neighbor searching in 3D, In arXiv:2311.01597.
[9] P. K. Agarwal and J. Matoušek, Ray shooting and parametric search, *SIAM J. Comput.* 22 (1993), 794–806.
[10] P. K. Agarwal and J. Matousek, On range searching with semialgebraic sets, *Discrete Comput. Geom.* 11 (1994), 393–418.
[11] P. K. Agarwal, J. Matousek, O. Schwarzkopf, Computing many faces in arrangements of lines and segments, *SIAM J. Comput.* 27(2) (1998), 491–505.
[12] P. K. Agarwal, J. Matoušek, and M. Sharir, On range searching with semialgebraic sets II, *SIAM J. Comput.* 42 (2013), 2039–2062.
[13] P. K. Agarwal and M. Sharir, Efficient randomized algorithms for some geometric optimization problems, *Discrete Comput. Geom.* 16 (1996), 317–337.
[14] P. K. Agarwal and M. Sharir, Arrangements of surfaces in higher dimensions, in *Handbook of Computational Geometry* (eds. J.R. Sack and J. Urrutia), pp. 49–119, North-Holland, Amsterdam, 2000.
[15] P. K. Agarwal and M. Sharir, Pipes, cigars, and kreplach: The union of Minkowski sums in three dimensions, *Discrete Comput. Geom.* 24 (2000), 645–685.
[16] P. K. Agarwal, M. Sharir and A. Steiger, Decomposing the complement of the union of cubes in three dimensions, *Proc. 32nd Annu. ACM-SIAM Sympos. Discrete Algorithms*, 2021, 1425–1444.
[17] N. Alon, and J. H. Spencer, *The Probabilistic Method*, Third Edition. Wiley-Interscience series in discrete mathematics and optimization, Wiley, New York, 2008.
[18] B. Aronov, A. Efrat, V. Koltun, and M. Sharir, On the union of κ-round objects in three and four dimensions, *Discrete Comput. Geom.* 36(4) (2006), 511–526.
[19] B. Aronov and M. Sharir, Triangles in space or building (and analyzing) castles in the air, *Combinatorica* 10(2) (1990), 137–173.
[20] S. Basu, R. Pollack, and M.-F. Roy, *Algorithms in Real Algebraic Geometry*, 2nd Edition, Springer Verlag, Berlin, 2006.
[21] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd Ed., Springer Verlag, Berlin-Heidelberg, 2008.
[22] M. de Berg, L. J. Guibas and D. Halperin, Vertical decomposition for triangles in 3-space, *Discrete Comput. Geom.* 15 (1996), 35–61.
[23] M. de Berg and O. Schwarzkopf, Cuttings and applications, *Int. J. Comput. Geom. Appl.* 5(4) (1995), 343–355.
[24] B. Chazelle, H. Edelsbrunner, L. Guibas and M. Sharir, A singly exponential stratification scheme for real semi-algebraic varieties and its applications, *Theoret. Comput. Sci.* 84 (1991), 77–105. Also in *Proc. 16th Int. Colloq. on Automata, Languages and Programming*, 1989, pp. 179–193.
[25] B. Chazelle and J. Friedman, A deterministic view of random sampling and its use in geometry, *Combinatorica*, 10 (1990), 229–249.
[26] B. Chazelle and J. Incerpi, Triangulation and shape-complexity, *ACM Trans. Graphics* 3 (1984), 135–152.
[27] K. L. Clarkson, New applications of random sampling in computational geometry, *Discrete Comput. Geom.*, 2 (1987), 195–222.
[28] K. L. Clarkson, H. Edelsbrunner, L. J. Guibas, M. Sharir, and E. Welzl, Combinatorial complexity bounds for arrangement of curves and spheres, *Discrete Comput. Geom.* 5 (1990), 99–160.
[29] G. E. Collins, Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition, *Proc. 2nd GI Conf. Automata Theory and Formal Languages*, volume 33. Springer LNCS, 1975.
[30] E. Ezra, On the union of cylinders in three dimensions, *Discrete Comput. Geom.* 45(1) (2011), 45–64.
[31] E. Ezra and M. Sharir, On the union of fat tetrahedra in three dimensions, *J. ACM* 57(1) (2009), 2:1–2:23.
[32] L. J. Guibas, D. Halperin, J. Matousek, and M. Sharir, Vertical decomposition of arrangements of hyperplanes in four

dimensions, *Discrete Comput. Geom.*, 14(2) (1995), 113–122.
[33] D. Halperin and M. Sharir, A near-quadratic algorithm for planning the motion of a polygon in a polygonal environment, *Discret. Comput. Geom.* 16(2) (1996), 121–134.
[34] S. Har-Peled, Multicolor combination lemma, *Comput. Geom.* 12 (1999), 155–176.
[35] D. Haussler and E. Welzl, Epsilon-nets and simplex range queries, *Discrete Comput. Geom.*, 2 (1987), 127–151.
[36] V. Koltun, Almost tight upper bounds for vertical decompositions in four dimensions, *J. ACM* 51(5) (2004), 699–730.
[37] V. Koltun, Sharp bounds for vertical decompositions of linear arrangements in four dimensions, *Discrete Comput. Geom.* 31(3) (2004), 435–460.
[38] V. Koltun and M. Sharir, The partition technique for the overlay of envelopes. *SIAM J. Comput.* 32 (2003), 841–863.
[39] J. Matoušek, Efficient partition trees, *Discrete Comput. Geom.*, 8 (1992), 315–334.
[40] J. Matoušek, Reporting points in halfspaces, *Comput. Geom. Theory Appl.* 2 (1992), 169–186.
[41] J. Matoušek and Z. Patáková, Multilevel polynomial partitioning and simplified range searching, *Discrete Comput. Geom.* 54 (2015), 22–41.
[42] S. Mohaban and M. Sharir, Ray shooting amidst spheres in 3 dimensions and related problems. *SIAM J. Comput.* 26 (1997), 654–674.
[43] J. T. Schwartz and M. Sharir, On the Piano Movers' problem: II. General techniques for computing topological properties of real algebraic manifolds, *Advances in Appl. Math.*, 4 (1983), 298–351.
[44] O. Schwarzkopf and M. Sharir, Vertical decomposition of a single cell in a three-dimensional arrangement of surfaces, *Discrete Comput. Geom.*, 18(3) (1997), 269–288.
[45] M. Sharir and P.K. Agarwal, *Davenport-Schinzel Sequences and Their Geometric Applications*, Cambridge University Press, Cambridge-New York-Melbourne, 1995.
[46] M. Sharir and H. Shaul, Semialgebraic range reporting and emptiness searching with applications, *SIAM J. Comput.* 40(4) (2011), 1045–1074.
[47] B. Tagansky, A new technique for analyzing substructures in arrangements of piecewise linear surfaces, *Discret. Comput. Geom.* 16(4) (1996), 455–479.