Computing a Center-Transversal Line*

Pankaj K. Agarwal [1] Sergio Cabello [2] J. Antoni Sellarès [3] Micha Sharir [4]

[1] Department of Computer Science, Duke University, P.O. Box 90129, Durham NC 27708-0129, USA, pankaj@cs.duke.edu
[2] Department of Mathematics, IMFM, and Department of Mathematics, FMF, University of Ljubljana, Slovenia, sergio.cabello@fmf.uni-lj.si
[3] Institut d'Informàtica i Aplicacions, Universitat de Girona, Spain, sellares@ima.udg.es
[4] School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel, and Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, USA, michas@post.tau.ac.il

Abstract

A center-transversal line for two finite point sets in $\mathbb{R}^3$ is a line with the property that any closed halfspace that contains it also contains at least one third of each point set. It is known that a center-transversal line always exists,but the best known algorithm for finding such a line takes roughly n^{12} time. We propose an algorithm that finds a center-transversal line in $O(n^{1+\varepsilon}\kappa^2(n))$ worst-case time, for any $\varepsilon > 0$, where $\kappa(n)$ is the maximum complexity of a single level in an arrangement of n planes in $\mathbb{R}^3$. With the current best upper bound $\kappa(n) = O(n^{5/2})$, the running time is $O(n^{6+\varepsilon})$, for any $\varepsilon > 0$. We also show that the problem of deciding whether there is a center-transversal line parallel to a given direction can be solved in $O(n \log n)$ expected time. Finally, we extend the concept of center-transversal line to that of bichromatic depth of lines in space, and give an algorithm that computes a deepest line exactly in time $O(n^{1+\varepsilon}\kappa^2(n))$, and a linear-time approximation algorithm that computes, for any specified $\delta > 0$, a line whose depth is at least $1 - \delta$ times the maximum depth.

*P.A. was supported by NSF under grants CCR-00-86013 EIA-98-70724, EIA-99-72879, EIA-01-31905, and CCR-02-04118. S.C. was partially supported by the European Community Sixth Framework Programme under a Marie Curie Intra-European Fellowship, and by the Slovenian Research Agency, project J1-7218. J.A.S. was partially supported by grant TIN2004-08065-C02-02 of the Spanish Ministry of Education and Science (MEC). M.S. was partially supported by NSF Grants CCR-00-98246 and CCF-05-14079, by grant 155/05 of the Israel Science Fund, and by the Hermann Minkowski–MINERVA Center for Geometry at Tel Aviv University. P.A. and M.S. were also supported by a joint grant from the U.S.-Israeli Binational Science Foundation.
A preliminary version of the paper had appeared in the *Proceedings of the Twenty-Sixth International Conference on Foundations of Software Technology and Theoretical Computer Science, 2006, pp. 93–104.*

1 Introduction

Center points and ham-sandwich cuts are two classical notions in discrete geometry. Given a set P of points in $\mathbb{R}^d$, a point q, not necessarily in P, is a *center point* with respect to P if any closed halfspace that contains q also contains at least $|P|/(d+1)$ points of P. The existence of center points is a consequence of Helly's theorem [20]. Given d finite point sets $P_0, \ldots, P_{d-1}$ in $\mathbb{R}^d$ with n points in total, a *ham-sandwich cut* is a hyperplane h such that each of the open halfspaces bounded by h contains at most $|P_i|/2$ points of P_i, for every $i = 0, 1, \ldots, d-1$. Dol'nikov [15], and Živaljević and Vrećica [31] proved the following theorem, called *center-transversal theorem*, which yields a generalization of center points and ham-sandwich cuts.

Theorem 1.1 (Center-Transversal Theorem) *Given $k+1$ finite point sets $P_0, P_1, \ldots, P_k$ in $\mathbb{R}^d$, for any $0 \le k \le d-1$, there exists a k-flat f such that any closed halfspace that contains f also contains at least $\frac{1}{d-k+1}|P_i|$ points of P_i, for each $i = 0, 1, \ldots, k$.*

Observe that when $k = 0$, f is a center point, and when $k = d-1$, f is a ham-sandwich cut. Therefore, the center-transversal theorem can be seen as an "interpolation" between these two theorems. A weaker result with $|P_i|/(d+1)$ instead of $|P_i|/(d-k+1)$ can easily be obtained by considering the k-flat passing through a center point of each of the P_i, $i = 0, 1, \ldots, k$.

In this paper we consider in detail the case $d = 3$, $k = 1$. Given two finite point sets P_0, P_1 in $\mathbb{R}^3$, we say that a line ℓ is a *center-transversal line* for P_0, P_1 if any closed half-space that contains ℓ also contains at least $|P_i|/3$ points of P_i, for $i = 0, 1$. The center-transversal theorem asserts that, for any finite point sets P_0, P_1 in $\mathbb{R}^3$, there exists a center-transversal line. However, the original proofs [15, 31] of this result are non-constructive and do not lead to an algorithm for finding a center-transversal line. The running time of the best known algorithm for this problem [5] is rather large (about n^{12}—see below). We present a considerably more efficient algorithm for finding such a line, and consider several other related problems.

Related work. A more detailed review of center points, ham sandwich cuts, and related problems can be found in Matoušek [20]. Efficient algorithms are known for computing a center point in $\mathbb{R}^2$ and $\mathbb{R}^3$ [13, 18, 21]. A center point in $\mathbb{R}^d$ can be found using linear programming with $\Theta(n^d)$ linear inequalities, and there exists a faster algorithm, due to Clarkson et al. [12], for computing an *approximate* center point in arbitrary dimensions; that is, a point q such that any closed halfspace containing q contains at least $\Omega(n/d^2)$ points of P. Efficient algorithms have also been developed for constructing the *center region*, namely, the set of all center points, in $\mathbb{R}^2$ and $\mathbb{R}^3$ [4, 7, 19]. The concept of center point leads to generalizations that have been useful in robust statistics. The *halfspace depth* (also called location depth, data depth) of a point q relative to a data set P in $\mathbb{R}^d$, is the smallest number of data points in any closed halfspace whose boundary passes through q. A center point is a point with depth at least $|P|/(d+1)$, and a halfspace median, or a *Tukey point*, is a point with maximum halfspace depth. Chan [7], improving upon previous results, has obtained a randomized $O(n \log n + n^{d-1})$ expected-time algorithm for computing a Tukey point in $\mathbb{R}^d$.

The problem that we consider can be related to *multivariate regression depth*, a generalization, introduced by Bern and Eppstein [5], of *regression depth*, a quality measure for robust linear regression defined by Rousseeuw and Hubert [17, 26, 27]. In particular, Bern and Eppstein [5] give a general-purpose algorithm, which can be easily modified to yield an algorithm that constructs a center-transversal line in $\mathbb{R}^3$ in $O(n^{12+\varepsilon})$ time, for any $\varepsilon > 0$.

Our contributions. Let P_0, P_1 be two finite point sets in $\mathbb{R}^3$ with a total of n points.

- We present an algorithm that constructs a center-transversal line for P_0 and P_1 in $O(n^{1+\varepsilon}\kappa^2(n))$ worst-case time, for any $\varepsilon > 0$, where $\kappa(n)$ is the maximum complexity of a single level in an arrangement of n planes in $\mathbb{R}^3$. With the current best upper bound $\kappa(n) = O(n^{5/2})$ of [28], the running time is $O(n^{6+\varepsilon})$, for any $\varepsilon > 0$. This is a considerable improvement over the algorithm by Bern and Eppstein [5].[1] This improvement is attained by analyzing the combinatorial structure of the problem, by searching for candidate center-transversal lines in a controlled recursive manner, and by using (standard) range-searching data structures for handling the interaction between lines and polyhedral terrains. See Section 2.

- Using a simple relation between center-transversal lines and center points in two dimensions, we show how to decide in $O(n \log n)$ time, for a given direction, whether there exists a center-transversal line of P_0 and P_1 with that direction. See Section 3.

- We introduce the notion of the *bichromatic depth* of a line ℓ, with respect to P_0 and P_1, extending similar earlier concepts. Specifically, it is the minimum fraction size ρ of the points in either set that lie in a halfspace that contains ℓ; that is, each halfspace containing ℓ contains at least $\rho|P_0|$ points of P_0 and $\rho|P_1|$ points of P_1. This concept generalizes that of center-transversal line (which has bichromatic depth at least $1/3$). We show how to compute a deepest line in $O(n^{1+\varepsilon}\kappa^2(n))$ time, for any $\varepsilon > 0$, and give a linear-time approximation algorithm that computes, for any $\delta > 0$, a line whose depth is at least $1-\delta$ times the maximum depth. See Section 4.

2 Finding a Center-Transversal Line

We consider the problem of computing a center-transversal line in dual space, where the problem is reformulated in terms of levels in arrangements of planes. We generate a set of candidate lines that is guaranteed to contain a center-transversal line, and we use a data structure to determine which of these candidate lines is a center-transversal line. For simplicity, we assume that $P_0 \cup P_1$ are in general position in the sense that no four of them are coplanar.

Center-transversal lines in the dual. The widely used *duality* transform maps a point p in $\mathbb{R}^d$ to a hyperplane p^* in $\mathbb{R}^d$ and vice-versa, so that the incidence and above/below relationships are preserved. There are many variants of duality [20]; we use the following one: A point $a = (a_1, \ldots, a_d) \in \mathbb{R}^d$ is mapped to the nonvertical hyperplane $a^* : x_d = a_1x_1 + \cdots + a_{d-1}x_{d-1} - a_d$, and a hyperplane $h : x_d = \alpha_1x_1 + \cdots + \alpha_{d-1}x_{d-1} + \alpha_d$ is mapped to the point $h^* =$

[1] We note though that an algorithm with running time near n^8 is not hard to obtain.

$(\alpha_1, \ldots, \alpha_{d-1}, -\alpha_d)$, so $(a^*)^* = a$. A point p lies below (resp., above, on) a hyperplane h if the dual point h^* lies below (resp., above, on) the dual hyperplane p^*. The *pencil* of hyperplanes passing through a line ℓ in $\mathbb{R}^d$, for $d \geq 3$, maps to the set of points in $\mathbb{R}^d$ lying on a line ℓ^*; we refer to ℓ^* as the dual of ℓ. For a set A of objects, set $A^* = \{a^* \mid a \in A\}$.

Let P be a set of n points in $\mathbb{R}^3$, and let $H = P^*$ be the set of n non-vertical planes in $\mathbb{R}^3$ dual to the points in P. The *level* of a point $p \in \mathbb{R}^3$, with respect to H, is the number of planes in H that lie *below* p. For $0 \leq k < n$, the *k-level* of H, denoted $\mathcal{L}_k(H)$ (or simply $\mathcal{L}_k$ if the set H is understood), is the closure of the set of all points on any of the planes of H that are at level k. The k-level $\mathcal{L}_k$ is a *polyhedral terrain*, that is, an xy-monotone piecewise-linear continuous surface formed by a subset of the faces of the arrangement $\mathcal{A}(H)$. The combinatorial complexity of $\mathcal{L}_k$ is the number of faces of all dimensions in $\mathcal{L}_k$. Let $\kappa(n)$ denote the maximum complexity of a level in any arrangement of n planes in $\mathbb{R}^3$. The best known upper bound for $\kappa(n)$ is $O(n^{5/2})$ [28], which differs substantially from the best known lower bound $n^2 e^{\Omega(\sqrt{\log n})}$ [30, 22]. See [3] for more details on arrangements and levels.

If h is a plane in $\mathbb{R}^3$ so that each of the two halfspaces bounded by h contains at least k points of P, then h^* lies between $\mathcal{L}_k(H)$ and $\mathcal{L}_{n-k}(H)$. If ℓ is a line in $\mathbb{R}^3$ so that any halfspace containing ℓ contains at least k points of P, then the entire dual line ℓ^* lies between $\mathcal{L}_k(H)$ and $\mathcal{L}_{n-k}(H)$. Hence, the problem of computing a center-transversal line for P_0 and P_1 reduces to computing a line in the dual space that lies above $\Sigma_0 = \mathcal{L}_{k_0}(H_0), \Sigma_1 = \mathcal{L}_{k_1}(H_1)$ and below $\Sigma_2 = \mathcal{L}_{n_0-k_0}(H_0), \Sigma_3 = \mathcal{L}_{n_1-k_1}(H_1)$, where $H_i = P_i^*$, $n_i = |P_i|$, and $k_i = \lceil n_i/3 \rceil$ for $i = 0, 1$. We note that each of these four terrains can be computed in $O(n^\varepsilon \kappa(n))$ time, for any $\varepsilon > 0$ [2].

We thus have four terrains $\Sigma_0, \Sigma_1, \Sigma_2, \Sigma_3$, and we wish to compute a line that lies above Σ_0, Σ_1 and below Σ_2, Σ_3. Note that such a line cannot be *z-vertical*, i.e., parallel to the z-axis. Let E_i be the set of edges in Σ_i, for $i = 0, 1, 2, 3$, and $E = \bigcup_{i=0}^3 E_i$. Set $m := |E| \leq 4\kappa(n)$, and assume that $m \geq n$ (or else the problem can be solved much faster than the time bound of our algorithm). Let $H = H_0 \cup H_1$. Each edge in E_i lies in the intersection line of a pair of planes in H. We define a "sidedness function" $\chi : E \to \{+1, -1\}$, where $\chi(e) = +1$ if $e \in E_0 \cup E_1$ and $\chi(e) = -1$ if $e \in E_2 \cup E_3$. Let V be the set of endpoints of edges in E. By the general-position assumption on input points, each point of V is incident upon at most three edges of E. Note that the edges in E are *not* in general position because many of them can be collinear or coplanar. For an object (point, line, segment) Δ in $\mathbb{R}^3$, let $\tilde{\Delta}$ denote its xy-projection in $\mathbb{R}^2$.

Definition 2.1 Let ℓ be a nonvertical line in $\mathbb{R}^3$, and let e be a nonvertical segment in $\mathbb{R}^3$ so that $\tilde{\ell}$ intersects $\tilde{e}$. We say that ℓ lies *above* (resp., *below*) e if the oriented line in the $(+z)$-direction that passes through $\tilde{\ell} \cap \tilde{e}$ meets e before (resp., after) ℓ. The line ℓ is in *compliance* with an edge $e \in E$ if (i) $\tilde{\ell}$ does not intersect $\tilde{e}$, or (ii) ℓ does not lie below (resp., above) e if $\chi(e) = +1$ (resp., $\chi(e) = -1$). We say that ℓ is in compliance with a subset $R \subseteq E$ if it is in compliance with every edge in R. In particular, we have:

Lemma 2.2 *A nonvertical line ℓ in $\mathbb{R}^3$ lies above Σ_0, Σ_1 and below Σ_2, Σ_3 if and only if ℓ is in compliance with E.*

The problem of computing a center-transversal line now reduces to finding a line that is in compliance with E. Let $\mathbb{L}$ be the set of all lines in $\mathbb{R}^3$ that are not parallel to the yz-plane, and let $\mathbb{L}_{yz}$ be the set of lines in $\mathbb{R}^3$ that are parallel to the yz-plane. We restrict the search for a line that is in compliance with E to lines in $\mathbb{L}$. This involves no loss of generality: The lines in $\mathbb{L}_{yz}$ have three degrees of freedom and a center-transversal line among them, if there exists one, can be found using a much simpler (and more efficient) algorithm; see, e.g., the remark following Lemma 2.7.

Overview of the algorithm. Before describing the algorithm in detail, we give a brief overview of the algorithm. Using the fact that each terrain Σ_i is contained in the union of n planes, we show that, for each line $\ell \in \mathbb{L}$, there exists a "witness set" of $O(n)$ edges of E, so that ℓ is in compliance with E if and only if it is in compliance with its witness set. The concept of witness sets is the basic tool to obtain an improved running time over trivial algorithms. We then group the lines in $\mathbb{L}$ into equivalence classes so that all lines in the same class have the same witness set.

Using these ideas, we present an algorithm to construct a set of candidate lines that works in three stages. The first stage, called the *filtering stage*, splits the problem into $O(m^2/n^2)$ subproblems (recall that m is the number of edges in our four terrains). Each subproblem is defined by a triangle Δ in a parametric plane, and corresponds to the set of candidate lines whose xy-projection dualizes (in the plane) to a point in Δ. In each subproblem there is a witness set of $O(n)$ edges, and therefore we obtain $O(m^2/n^2)$ subproblems, each involving only $O(n)$ edges. The second stage, called the *recursive candidate generation stage*, computes, for each subproblem, a set of $O(n^{3+\varepsilon})$ candidate lines, for any $\varepsilon > 0$, which is guaranteed to contain a line in compliance with the corresponding subset of edges if there exists one. This stage is similar to the approach used in [23, 25]. The final stage, called the *verification stage*, uses known data structures to check which of the candidate lines generated by the previous step is in compliance with E, and report the first such line that it encounters (which is guaranteed to exist, by Theorem 1.1). We now describe each of these steps in detail.

Witness sets and equivalence classes. For a line $\ell \in \mathbb{L}$ and a subset $R \subseteq E$ of edges, we define the *witness set* of ℓ for R, denoted by $W(\ell, R)$, as follows. For $i = 0, 1, 2, 3$, let $R_i \subseteq R$ be the sequence of edges in $R \cap E_i$ whose xy-projections intersect $\tilde{\ell}$, sorted by the order of the intersection points along $\tilde{\ell}$. For a plane $h \in H_0 \cup H_1$, let $e^-_{h,i}, e^+_{h,i} \in R_i$ be, respectively, the first and the last edges in the i-th sequence that lie on h, where only planes in H_0 (resp., H_1) are considered for $i = 0, 2$ (resp., $i = 1, 3$). We set

$$W(\ell, R) = \{e^-_{h,i}, e^+_{h,i} \mid h \in H,\ 0 \le i \le 3\}.$$

By definition, $\tilde{\ell}$ intersects the xy-projection of every edge in $W(\ell, R)$. Note that $|W(\ell, R)| = O(n)$.

Lemma 2.3 *For a subset $R \subseteq E$, a line $\ell \in \mathbb{L}$ is in compliance with R if and only if ℓ is in compliance with $W(\ell, R)$.*

The proof of the lemma follows from the simple observation that if ℓ lies above (resp., below) both $e^-_{h,i}, e^+_{h,i}$ then it lies above (resp., below) all edges in R_i that lie in h.

We define, for a subset $R \subseteq E$, an equivalence relation on $\mathbb{L}$ so that for any two lines ℓ_1, ℓ_2 in the same equivalence class, $W(\ell_1, R) = W(\ell_2, R)$. This will discretize the search for a center-transversal line. For this we need a few notations. For a point, a line, or a segment ξ in $\mathbb{R}^3$, let $\varphi(\xi)$ denote the dual (in $\mathbb{R}^2$) of $\tilde{\xi}$, i.e., $\varphi(\xi) = (\tilde{\xi})^*$. [2] For an edge $e = uv$ in E, let $\varphi(e) \subseteq \mathbb{R}^2$ be the double wedge that is formed by the lines $\varphi(u)$ and $\varphi(v)$ and does not contain the line in $\mathbb{R}^2$ passing through their intersection point and parallel to the y-axis. By standard properties of the duality transform in $\mathbb{R}^2$, a line γ in $\mathbb{R}^2$ intersects $\tilde{e}$ if and only if $\gamma^* \in \varphi(e)$. Moreover if the points $\gamma_1^*, \gamma_2^* \in \mathbb{R}^2$ lie in the same (left or right) wedge of $\varphi(e)$, then γ_1, γ_2 intersect $\tilde{e}$ from the *same side*, in the sense that the same endpoint of $\tilde{e}$ lies in each of the positive halfplanes bounded by γ_1 and γ_2, respectively (that is, the halfplanes above these lines, in the y-direction).

Let $R \subseteq E$ be a fixed subset of edges, and let $V_R \subseteq V$ be the set of endpoints of the edges in R. For a point $v \in V_R$, $\varphi(v)$ is the line in $\mathbb{R}^2$ dual to the point $\tilde{v}$. Set $\Lambda(R) = \{\varphi(v) \mid v \in V_R\}$. For each face f in the arrangement $\mathcal{A}(\Lambda(R))$ of $\Lambda(R)$, let $R(f)$ denote the set of those edges $e \in R$ for which $\varphi(e)$ contains f. For a line $\ell \in \mathbb{L}$, if f is the face containing $\varphi(\ell)$ then, by construction, $R(f)$ is the set of edges of R whose xy-projections intersect $\tilde{\ell}$. By definition, $W(\ell, R) \subseteq R(f)$. We note that, in general, $\Lambda(R)$ is not in general position, since it consists of lines dual to points that lie on $O(n^2)$ lines (namely, the projections of the intersection lines between pairs of planes in H_0 or in H_1). Nevertheless, the techniques that we are about to apply (such as cuttings of arrangements) work equally well in degenerate scenarios.

Definition 2.4 We call two lines $\ell_1, \ell_2 \in \mathbb{L}$ *equivalent* (with respect to R), denoted by $\ell_1 \equiv_R \ell_2$, if $\varphi(\ell_1)$ and $\varphi(\ell_2)$ lie in the same face of $\mathcal{A}(\Lambda(R))$.

Lemma 2.5 *Let $R \subseteq E$ be a set of edges, and let $\ell_1, \ell_2 \in \mathbb{L}$ be two lines so that $\ell_1 \equiv_R \ell_2$. Then $W(\ell_1, R) = W(\ell_2, R)$.*

Proof: Let f be the face of $\mathcal{A}(\Lambda(R))$ that contains $\varphi(\ell_1)$ and $\varphi(\ell_2)$. Set $R_i(f) := R(f) \cap E_i$ and $L_i := \Lambda(R_i(f)) \subseteq \Lambda(R)$, for $i = 0, 1, 2, 3$. Clearly, $\varphi(\ell_1), \varphi(\ell_2)$ lie in the same face of $\mathcal{A}(L_i)$. Since the edges of E_i all belong to the same terrain, their xy-projections are pairwise disjoint. An easy observation (due to [1]) shows that $\tilde{\ell_1}, \tilde{\ell_2}$ intersect the xy-projections of the edges in $R_i(f)$ in the same order. This immediately implies that $W(\ell_1, R) \cap E_i = W(\ell_2, R) \cap E_i$, from which the lemma follows. $\Box$

In view of the preceding lemma, we define, for each face f of $\mathcal{A}(R)$, $W_f(R) \subseteq R$ to be the common witness set for any line in the equivalence class corresponding to f.

The filtering stage. Given a set L of lines in $\mathbb{R}^2$, a triangle Δ_0 (crossed by all the lines in L), and a parameter $1 \le r \le |L|$, a $(1/r)$-*cutting* of (L, Δ_0) is a triangulation Ξ of Δ_0 so that each triangle of Ξ is crossed by at most $|L|/r$ lines of L. It is known that a $(1/r)$-cutting consisting of $O(r^2)$ triangles, along with the set of lines crossing each of its triangles, can be computed in $O(|L|r)$ time [9].

[2] Note that $\varphi(\ell)$ is not defined if ℓ is parallel to the yz-plane. That is why we exclude these lines from $\mathbb{L}$.

Let $\Lambda = \Lambda(E)$. We set $\Delta_0 = \mathbb{R}^2$ and $r = m/n$, and compute a $(1/r)$-cutting Ξ of (Λ, Δ_0). For each triangle $\Delta \in \Xi$, let Λ_Δ be the set of lines of Λ that cross Δ; since Ξ is a $(1/r)$-cutting, we have $|\Lambda_\Delta| \le m/r = n$. Let $E_\Delta \subseteq E$ be the set of edges $e = uv$ so that either $\varphi(u)$ or $\varphi(v)$ belongs to Λ_Δ. Since each vertex of V is an endpoint of at most three edges of E, we have $|E_\Delta| \le 3|\Lambda_\Delta| \le 3n$. For each $\Delta \in \Xi$, let $F_\Delta = \{e \in E \setminus E_\Delta \mid \Delta \subseteq \varphi(e)\}$. We refer to the edges in E_Δ as *short* and to the edges in F_Δ as *long* (in Δ). Finally, let $\mathbb{L}_\Delta = \{\ell \in \mathbb{L} \mid \varphi(\ell) \in \Delta\}$.

Since Δ is contained in a face of $\mathcal{A}(\Lambda(F_\Delta))$ (the arrangement of lines dual to the xy-projections of the endpoints of the edges in F_Δ), Lemma 2.5 implies that $W(\ell, F_\Delta)$ is the same for all lines $\ell \in \mathbb{L}_\Delta$; let W_Δ denote this common witness set. Observe that $|W_\Delta| = O(n)$.

If two triangles Δ and Δ' in Ξ share an edge, then the symmetric difference of $F_\Delta, F_{\Delta'}$ is a subset of $E_\Delta \cup E_{\Delta'}$. Therefore W_Δ can be computed from $W_{\Delta'}$ in $O(|E_\Delta| + |E_{\Delta'}|) = O(n)$ time. Hence, by performing a traversal of Ξ, we can compute W_Δ for all triangles $\Delta \in \Xi$, in overall time $O(m^2/n)$.

The next lemma follows from Lemmas 2.3 and 2.5.

Lemma 2.6 *For any $\Delta \in \Xi$, a line $\ell \in \mathbb{L}_\Delta$ is in compliance with E if and only if ℓ is in compliance with $E_\Delta \cup W_\Delta$.*

Hence, for each $\Delta \in \Xi$, we have a subproblem $(\Delta, E_\Delta, W_\Delta)$, in which we want to determine whether there is a line in $\mathbb{L}_\Delta$ that is in compliance with $E_\Delta \cup W_\Delta$ (and thus with E). Since $\bigcup_\Delta \mathbb{L}_\Delta = \mathbb{L}$, these subproblems together exhaust the overall problem of computing a line in $\mathbb{L}$ that is in compliance with E. There are $O(m^2/n^2)$ such subproblems, and the total time spent in generating them is $O(m^2/n)$.

The recursive candidate generation stage. Let $(\Delta, E_\Delta, W_\Delta)$ be one of the subproblems generated in the previous stage. We generate a set of "candidate" lines that is guaranteed to contain a line in compliance with $E_\Delta \cup W_\Delta$ if there exists one in $\mathbb{L}_\Delta$. If there is a line in $\mathbb{L} \setminus \mathbb{L}_\Delta$ in compliance with $E_\Delta \cup W_\Delta$, then the candidate set may or may not contain such a line. Moreover, a candidate line generated by the algorithm may be parallel to the yz-plane. The time used to generate this set of "candidate" lines will be $O(n^{3+\varepsilon})$, for any $\varepsilon > 0$.

Before describing the algorithm, we briefly review the representation of lines in Plücker space [24]. An oriented line ℓ in $\mathbb{R}^3$ can be mapped to a point $\pi(\ell) \in \mathbb{R}^5$, called the *Plücker point* of ℓ, that lies on the so-called 4-dimensional *Plücker hypersurface* Π, or to a hyperplane $\varpi(\ell)$ in $\mathbb{R}^5$, called the *Plücker hyperplane* of ℓ. (The actual Plücker space is the *real projective* $\mathbb{R}^5$, but since we exclude lines parallel to the yz-plane, one of the homogeneous coordinates is always nonzero, and hence we can embed the Plücker structure into the real 5-dimensional space. However, the mapping $\varpi(\cdot)$ is also defined for lines parallel to the yz-plane.) Abusing the notation a little, we use $\pi(e)$ and $\varpi(e)$ to denote the Plücker point and hyperplane, respectively, of the line supporting an oriented segment e in $\mathbb{R}^3$. Two lines ℓ_1, ℓ_2, where $\ell_1 \in \mathbb{L}$, are incident (or parallel) if and only if $\pi(\ell_1) \in \varpi(\ell_2)$.

We orient every line of $\mathbb{L}$ and every edge of E in the $(+x)$-direction (this is well defined for lines in $\mathbb{L}$, by definition, and for edges of E, by making a small rotation, if necessary). For two oriented lines ℓ_1, ℓ_2 in $\mathbb{R}^3$, $\pi(\ell_1)$ lies above $\varpi(\ell_2)$ (which is the same as $\pi(\ell_2)$ lying above $\varpi(\ell_1)$) if and only if the simplex spanned by a vector $\vec{u}_1$ lying on ℓ_1 with the same orientation, and by a vector $\vec{u}_2$ lying on ℓ_2 with the same orientation, is positively oriented. This is easily seen to imply that, when ℓ_1 and ℓ_2 are non-vertical, ℓ_1 passes above ℓ_2 if and only if either (i) $\pi(\ell_1)$ lies above $\varpi(\ell_2)$ and $\tilde{\ell_1}$ lies counterclockwise to $\tilde{\ell_2}$, or (ii) $\pi(\ell_1)$ lies below $\varpi(\ell_2)$ and $\tilde{\ell_1}$ lies clockwise to $\tilde{\ell_2}$. See [24] for more details.

Let η_1, η_2, η_3 be the vertical lines such that the lines dual to their intersections with the xy-plane support the respective edges e_1, e_2, e_3 of Δ. We also use the term "edges" when referring to η_1, η_2, η_3. For any line $\ell \in \mathbb{L}_\Delta$, let $B(\ell)$ denote the subset of edges of $X_\Delta = E_\Delta \cup W_\Delta \cup \{\eta_1, \eta_2, \eta_3\}$ that are touched by ℓ (in their relative interior or at an endpoint).

Assume now that there is a line $\ell \in \mathbb{L}_\Delta$ in compliance with $E_\Delta \cup W_\Delta$. If $\varphi(\ell)$ is a vertex of Δ, then ℓ touches two of the vertical edges of X_Δ. Since $\tilde{\ell}$ intersects an edge of each Σ_i, we can move ℓ in the vertical plane containing ℓ so that it touches two edges of $E_\Delta \cup W_\Delta$. Hence, if $\varphi(\ell)$ is a vertex, then we may assume that ℓ touhces at least four edges of X_Δ, otherwise we may assume that it touches at least two edges of X_Δ. Next, we move ℓ around while keeping it in compliance with $E_\Delta \cup W_\Delta$, keeping $\varphi(\ell)$ in Δ, and not losing any contact with an edge in X_Δ until we reach a critical position of ℓ at which $B(\ell)$ is maximal. We call a line in such position as a *critical line*. If $\varphi(\ell)$ reaches a boundary edge e_i of Δ during this motion, then ℓ touches the corresponding vertical line η_i. If Δ is unbounded, then ℓ may become parallel to the yz-plane. In other words, if there is a line $\ell \in \mathbb{L}_\Delta$ in compliance with $E_\Delta \cup W_\Delta$, then there is another line ℓ' such that ℓ' is in compliance with $E_\Delta \cup W_\Delta$, $\varphi(\ell) \in \Delta$ (possibly at infinity), $B(\ell) \subseteq B(\ell')$, and either $\ell' \in \mathbb{L}_\Delta$ is a critical line or $\ell \in \mathbb{L}_{yz}$. We focus on the case when $\ell' \in \mathbb{L}_\Delta$ and assume that ℓ' cannot be moved to infinity in the above motion without violating one of the constraints (see also the remark below).

Using the fact that a line in $\mathbb{R}^3$ has four degrees of freedom and the above argument, it can be verified that a critical line touches at least four edges of X_Δ. We next show that each critical line ℓ has a subset $A(\ell) \subseteq B(\ell)$ of four edges that defines ℓ, in the precise sense stated in the following lemma. This is easily seen to be the case when the segments in X_Δ are in general position. However, in our scenario many of these edges might be coplanar, so a more careful argument is needed.

Lemma 2.7 *If $\ell \in \mathbb{L}$ is a critical line, then there exists a subset $A(\ell) \subseteq B(\ell)$ consisting of four edges such that $\bigcap\{\varpi(e) \mid e \in A(\ell)\}$ is not contained in Π and $\pi(\ell)$ is one of the (at most two) points $\Pi \cap (\bigcap\{\varpi(e) \mid e \in A(\ell)\})$.*

Proof: Let $F(\ell) = \bigcap\{\varpi(e) \mid e \in B(\ell)\}$ be the flat in Plücker space that contains (the Plücker points of) all lines that meet $B(\ell)$.

We first claim that $F(\ell)$ cannot be a 1-flat contained in Π, nor can $F(\ell)$ contain a 2-flat. Indeed, in either of these cases $F(\ell)$ would contain (points representing) a 1-parameter continuous family of lines, including ℓ, so that all lines in this family touch $B(\ell)$. We can then move ℓ within this family until it touches another edge of X_Δ or becomes parallel to the yz-plane. In either case ℓ is not critical.

If $F(\ell)$ is a 1-flat not contained in Π, then $F(\ell)$ and the (quadratic) Plücker hypersurface Π intersect in at most two points. Moreover, there must exist four edges e_1, e_2, e_3, e_4 in $B(\ell)$ such that $F(\ell) = \bigcap\{\varpi(e) \mid e \in B(\ell)\} = \bigcap_{i=1}^4 \varpi(e_i)$. The subset $A(\ell) := \{e_1, \ldots, e_4\}$ has the desired properties.

If $F(\ell)$ is a 0-flat, then there must exist five edges $e_1, \ldots, e_5 \in B(\ell)$ such that $F(\ell) = \bigcap\{\varpi(e) \mid e \in B(\ell)\} = \bigcap_{i=1}^{5} \varpi(e_i)$. The Plücker hypersurface Π cannot contain the five different 1-flats obtained by intersecting any four of the hyperplanes $\varpi(e_1), \ldots, \varpi(e_5)$, because they span $\mathbb{R}^5$. Therefore, there exists a subset of four edges $A(\ell) := \{e'_1, \ldots, e'_4\} \subset \{e_1, \ldots, e_5\} \subseteq B(\ell)$ such that $\bigcap_{i=1}^{4} \varpi(e'_i)$ is a 1-flat not contained in Π and $\pi(\ell)$ is one of the (at most two) points $\Pi \cap \left(\bigcap_{i=1}^{4} \varpi(e'_i)\right)$. □

Remark. We can define the notion of "critical" lines for lines in $\mathbb{L}_{yz}$ as well. Since these lines have three degrees of freedom, an argument similar to the one in Lemma 2.7 shows that a critical line $\ell \in \mathbb{L}_{yz}$ is defined by a set $A(\ell) \subseteq B(\ell)$ of three segments in X_Δ. For each triple ξ of segments in X_Δ, we can compute the set of $O(1)$ "candidate" lines. The number of such candidate lines is $O(n^3)$, which we add to the overall set of candidate lines. In the rest of the section, we focus on generating critical lines in $\mathbb{L}_\Delta$.

For each critical line $\ell \in \mathbb{L}_\Delta$ we choose and fix a set $A(\ell)$ with the properties stated in Lemma 2.7, which contains as many edges from $\{\eta_1, \eta_2, \eta_3\}$ as possible. We classify each critical line $\ell \in \mathbb{L}_\Delta$ into one of the following types:

(E1) $A(\ell)$ contains two elements of $\{\eta_1, \eta_2, \eta_3\}$. In this case $\varphi(\ell)$ is a vertex of Δ.

(E2) $A(\ell)$ contains one element of $\{\eta_1, \eta_2, \eta_3\}$. In this case $\varphi(\ell)$ lies in the relative interior of an edge of Δ.

(E3) $A(\ell)$ is disjoint from $\{\eta_1, \eta_2, \eta_3\}$. In this case $\varphi(\ell)$ may lie anywhere in the interior of Δ.

Note that in cases (E2), (E3), $B(\ell)$ can contain additional edges η_i not included in $A(\ell)$. Since in cases (E1), (E2) $A(\ell)$ contains at most three edges of $E_\Delta \cup W_\Delta$, we can generate a set that contains all the critical lines of type (E1) or (E2) as follows: For each 4-tuple (e_1, e_2, e_3, e_4) of distinct edges, with $e_1 \in \{\eta_1, \eta_2, \eta_3\}$, $e_2 \in X_\Delta$, and $e_3, e_4 \in E_\Delta \cup W_\Delta$, we compute $\bigcap_{i=1}^{4} \varpi(e_i)$ and, if this space is a 1-flat not contained in Π, we add the (at most two) lines corresponding to $\Pi \cap \bigcap_{i=1}^{4} \varpi(e_i)$ to the candidate set. This procedure takes $O(n^3)$ time.

It remains to construct the set $\mathcal{C}(\Delta, E_\Delta, W_\Delta)$ of critical lines of type (E3). Each such line ℓ is in compliance with $E_\Delta \cup W_\Delta$, and is associated with a subset $A(\ell) \subseteq E_\Delta \cup W_\Delta$ of four edges such that $\bigcap\{\varpi(e) \mid e \in A(\ell)\}$ is a 1-flat not contained in Π and $\pi(\ell) \in \Pi \cap (\bigcap\{\varpi(e) \mid e \in A(\ell)\})$. We will compute a superset of $\mathcal{C}(\Delta, E_\Delta, W_\Delta)$ using a divide-and-conquer algorithm. Our approach for generating candidate lines is very similar to that used by Pellegrini [23] (see also [25]).

We choose a sufficiently large constant r, and construct a $(1/(6r))$-cutting T of $(\Lambda(E_\Delta), \Delta)$. As in the filtering stage, we define, for each $\tau \in T$, $E_\tau \subseteq E_\Delta$ to be the set of short edges in τ, and $F_\tau \subseteq E_\Delta$ to be the set of long edges in τ. We have $|E_\tau| \leq 3|\Lambda(E_\Delta)|/6r \leq |E_\Delta|/r$. Set $W_\tau := F_\tau \cup W_\Delta$. Define $\mathbb{L}_\tau = \{\ell \in \mathbb{L}_\Delta \mid \varphi(\ell) \in \tau\}$, and note that $\bigcup_{\tau \in T} \mathbb{L}_\tau = \mathbb{L}_\Delta$. For each $\tau \in T$, we compute a set of *candidate* lines $\mathcal{C}_\tau \subset \mathbb{L}_\tau$, with the property that $\mathcal{C}(\Delta, E_\Delta, W_\Delta) \subseteq \bigcup_{\tau \in T} \mathcal{C}_\tau$.

Consider a triangle $\tau \in T$. We want to construct a set of candidate lines $\mathcal{C}_\tau$ that includes the lines in $\mathbb{L}_\tau$ of type (E3). Hence, it suffices to consider only the edges $E_\tau \cup W_\tau$ in its construction. The line ℓ is in compliance with an edge $e \in W_\tau$ if $\pi(\ell)$ lies in one specific halfspace Γ_e bounded by $\varpi(e)$. Γ_e depends on the function $\chi(e)$ and on the clockwise order of $\tilde{\ell}$ and $\tilde{e}$ (when oriented in the positive x-direction). Since τ is a subset of a fixed wedge of $\varphi(e)$, this clockwise order is the same for all lines $\ell \in \mathbb{L}_\tau$; hence Γ_e is the same halfspace for all lines in $\mathbb{L}_\tau$. Set $\mathcal{K} := \bigcap_{e \in W_\tau} \Gamma_e$. $\mathcal{K}$ is a convex polyhedron in $\mathbb{R}^5$ with $O(n)$ facets, so its overall combinatorial complexity is $O(n^2)$, and it can be constructed in $O(n^2)$ time [8]. Note that if $\ell \in \mathbb{L}_\tau$ is a critical line in compliance with W_τ, then $\bigcap\{\varpi(e) \mid e \in A(\ell) \cap W_\tau\}$ supports a $(5-j)$-face of $\mathcal{K}$, where $j = |A(\ell) \cap W_\tau|$. There are four cases, depending on how many edges of W_τ are in $A(\ell)$.

$A(\ell) \subseteq W_\tau$. In this case the 1-flat $\bigcap\{\varpi(e) \mid e \in A(\ell)\}$ supports an edge of $\mathcal{K}$. Therefore, we find critical lines of this type as follows: For each edge of $\mathcal{K}$ that is not contained in Π, add the (at most) two lines corresponding to the intersection points of the edge and Π to the candidate set $\mathcal{C}_\tau$. The total time spent is $O(n^2)$.

$|A(\ell) \cap W_\tau| = 3$. In this case $\pi(\ell)$ lies in the intersection of the 2-flat $\{\varpi(e) \mid e \in A(\ell) \cap W_\tau\}$ with $\mathcal{K}$, so $\pi(\ell)$ lies on the intersection edge of some 2-face of $\partial\mathcal{K}$ and the Plücker hyperplane $\varpi(e)$ for some $e \in E_\tau$. We find critical lines of this type as follows: For each pair of an edge $e \in E_\tau$ and a 2-face ϕ of $\mathcal{K}$, compute the intersection $\phi \cap \varpi(e)$ and, if it is not contained in Π, add the (at most) two lines corresponding to the intersection points $\phi \cap \varpi(e) \cap \Pi$ to the candidate set $\mathcal{C}_\tau$. Since the polyhedron $\mathcal{K}$ has $O(n^2)$ 2-faces, the total number of lines generated in this case is $O(n^2|E_\tau|) = O(n^3/r)$, and their construction takes $O(n^3/r)$ time.

$|A(\ell) \cap W_\tau| = 2$. Let $e_1, e_2 \in E_\tau$ be the two edges that belong to $A(\ell)$. The Plücker subspace of lines (in $\mathbb{L}$) that touch e_1 and e_2 lies in the 3-dimensional flat $F = \varpi(e_1) \cap \varpi(e_2)$. Therefore, since ℓ is in compliance with W_τ, we have $\pi(\ell) \in F \cap \mathcal{K}$. Note that $F \cap \mathcal{K}$ is a convex 3-polyhedron with $O(n)$ facets, and therefore it has only $O(n)$ edges. We find critical lines ℓ of this type with $e_1, e_2 \in A(\ell)$, as follows: For each edge of $F \cap \mathcal{K}$ that is not contained in the Plücker surface Π, add the (at most two) lines corresponding to $F \cap \mathcal{K} \cap \Pi$ to the candidate set $\mathcal{C}_\tau$. This has to be done for each pair of edges $e_1, e_2 \in E_\tau$. The total number of lines generated in this case is $O(|E_\tau|^2 n) = O(n^3/r^2)$, and their computation takes $O(|E_\tau|^2 n \log n) = O((n^3/r^2)\log n)$ time, where the costliest step is the construction, repeated $O(|E_\tau|^2)$ times, of convex 3-polyhedra, each defined by at most n inequalities.

$|A(\ell) \cap W_\tau| \leq 1$. We partition W_τ (arbitrarily) into $u = O(r)$ subsets $W_\tau^{(1)}, \ldots, W_\tau^{(u)}$ so that $|W_\tau^{(i)}| \leq n/r$ for each i. We recursively compute the set of candidate lines $\mathcal{C}(\tau, E_\tau, W_\tau^{(i)})$, for $1 \leq i \leq u$ and for $\tau \in T$. We thus recursively solve $O(r)$ subproblems, all of whose outputs are added to our candidate set $\mathcal{C}_\tau$. Clearly, all lines of this type (and perhaps more) are found by this recursive procedure.

The correctness of the procedure is fairly straightforward. Let $T(n)$ denote the maximum time needed to compute $\bigcup_{\tau \in T} \mathcal{C}_\tau$, which is a superset of $\mathcal{C}(\Delta, E_\Delta, W_\Delta)$, when $|E_\Delta|, |W_\Delta| \leq n$. For each $\tau \in T$, we spend $O(n^2 + n^3/r + (n^3/r^2)\log n)$ time plus the time needed to solve $O(r)$ recursive calls where the size of each of the two sets of edges is at most n/r. Since the cutting T consists of $O(r^2)$ triangles, we obtain the following recurrence.

$$T(n) = O(r^3)T(n/r) + O(n^2 r^2 + n^3 r + n^3 \log n).$$

The solution of this recurrence is $T(n) = O(n^{3+\varepsilon})$, for any $\varepsilon > 0$ (for which we need to choose r sufficiently large, as a function of ε). The size of $\mathcal{C}(\Delta, E_\Delta, W_\Delta)$ is also bounded by this quantity.

Repeating this procedure for the $O(m^2/n^2)$ subproblems generated by the filtering stage and adding the set of critical lines in $\mathbb{L}_{yz}$, we construct, in $O(m^2 n^{1+\varepsilon})$ overall time, a set $\mathcal{C}$ of $O(m^2 n^{1+\varepsilon})$ candidate lines, guaranteed to contain a center-transversal line for P_0, P_1.

The verification stage. To complete the algorithm, we test which of the lines in $\mathcal{C}$ is in compliance with E. Using the data structure described in [11], we can preprocess, in $O(m^{2+\varepsilon})$ time, each E_i into a data structure of size $O(m^{2+\varepsilon})$ so that we can determine in $O(\log n)$ time whether a line $\ell \in \mathbb{L}$ passes above or below the terrain Σ_i, or, equivalently, whether ℓ is in compliance with E_i. Querying each line in $\mathcal{C}$ with this data structure for every E_i, we can determine, in $O(m^{2+\varepsilon} + m^2 n^{1+\varepsilon} \log n)$ time, which of the lines in $\mathcal{C}$ are in compliance with E. With an appropriate calibration of the parameters ε, we can rewrite this bound as $O(m^2 n^{1+\varepsilon})$, for any $\varepsilon > 0$.

Since a center-transversal line always exists, it belongs to $\mathcal{C}$, by construction, and will be found by this procedure. Putting everything together, and recalling that $m \le 4\kappa(n)$, where $\kappa(n)$ is the maximum complexity of a level in an arrangement of n planes in $\mathbb{R}^3$, we obtain the following main result of the paper. For the concrete time bound, we use the currently best known upper bound $\kappa(n) = O(n^{5/2})$ of [28].

Theorem 2.8 *A center-transversal line for two sets P_0, P_1 with a total of n points in $\mathbb{R}^3$ can be constructed in $O(n^{1+\varepsilon}\kappa^2(n))$ time, for any $\varepsilon > 0$. This bound is $O(n^{6+\varepsilon})$, for any $\varepsilon > 0$.*

Remarks. (1) It is strongly believed that $\kappa(n) = O(n^{2+\varepsilon})$, for any $\varepsilon > 0$, in which case our algorithm would take only $O(n^{5+\varepsilon})$ time, for any $\varepsilon > 0$.

(2) The general-position assumption on P_0 and P_1 was only used for bounding the size of E_Δ (or E_τ). If the points are not in general position, we can assign a *weight* to each line $\varphi(v) \in \Lambda(E)$, which is the number of edges of E incident upon v, and compute *weighted* $(1/r)$-cuttings [10], to obtain the same bound on the size of E_Δ (or E_τ).

(3) Let $\Psi(P_0, P_1)$ be the set of Plücker points of center transversal lines of P_0 and P_1. The above algorithm also proves that the combinatorial complexity of $\Psi(P_0, P_1)$ is $O(n^{1+\varepsilon}\kappa^2(n))$, for any $\varepsilon > 0$.

Terrains with many coplanar faces. Pellegrini [23] and Halperin and Sharir [16] have shown that the complexity of the envelope of lines lying above a polyhedral terrain of complexity k is $O(k^{3+\varepsilon})$, for any $\varepsilon > 0$. The complexity of this envelope is measured in terms of the number of critical lines that are tangent to the terrain while otherwise lying above it. In our scenario, we have taken advantage of the fact that the faces of our terrains are contained in relatively few planes. This observation can be extended to more general scenarios: Using the ideas of witness sets and the filtering stage, as in our analysis, we directly obtain the following result, which may be of independent interest.

Theorem 2.9 *Let Σ be a terrain of complexity k in $\mathbb{R}^3$, all of whose facets lie on n different planes. Then the complexity of the envelope of lines that pass above Σ is $O(n^{1+\varepsilon}k^2)$, for any $\varepsilon > 0$.*

3 Center-Transversal Line in a Given Direction

In this section we present a randomized algorithm for deciding whether there exists a center-transversal line of P_0 and P_1 in a given direction, say, the z-direction. Let $\tilde{P}_0$ (resp., $\tilde{P}_1$) be the xy-projection of P_0 (resp., P_1). A center-transversal line of P_0 and P_1 exists in the z-direction if and only if the intersection of the center regions of $\tilde{P}_0$ and $\tilde{P}_1$ is nonempty. Since each of these center regions can be computed in $O(n \log^2 n)$ randomized expected time [7] and their intersection can be computed in linear time, we can compute a center-transversal line in the z-direction, if it exists, in $O(n \log^2 n)$ expected time. Here we improve the expected running time to $O(n \log n)$.

For $i = 0, 1$, let H_i be the set of lines dual to $\tilde{P}_i$, and, for an integer k, let $\mathcal{L}_k(H_i)$ (resp., $\mathcal{U}_k(H_i)$) be the set of points whose level in $\mathcal{A}(H_i)$ is at most k (resp., at least $|H_i| - k$). In the dual setting, the problem of computing a center-transversal line in the z-direction reduces to determining whether there exists a line in the dual plane that lies above $\mathcal{L}_{k_0}(H_0) \cup \mathcal{L}_{k_1}(H_1)$ and below $\mathcal{U}_{k_0}(H_0) \cup \mathcal{U}_{k_1}(H_1)$, where $k_i = \lceil |P_i|/3 \rceil$ for $i = 0, 1$.

Let $\mathbb{L}$ be the set of all lines in $\mathbb{R}^2$. Suppose we have a (possibly infinite) set S of points in $\mathbb{R}^2$, in which each point is colored red or blue. We wish to compute

$$\begin{aligned} \omega(S) \quad := \quad & \min_{\ell \in \mathbb{L}} \ \mathrm{slope}(\ell) \\ & \text{s.t. } \ell \text{ lies above the red points of } S \\ & \text{and } \ell \text{ lies below the blue points of } S \end{aligned} \qquad (1)$$

As argued by Chan [7], this is an instance of linear programming. By setting $S = \mathcal{L}_k(H_0) \cup \mathcal{L}_k(H_1) \cup \mathcal{U}_k(H_0) \cup \mathcal{U}_k(H_1)$, where the points of $\mathcal{L}_k(H_0) \cup \mathcal{L}_k(H_1)$ are colored red and the points of $\mathcal{U}_k(H_0) \cup \mathcal{U}_k(H_1)$ are colored blue, we can reduce our problem to an instance of (1). Although the set S is infinite in our case, it suffices to consider the vertices of $\mathcal{L}_k(H_i)$ and $\mathcal{U}_k(H_i)$, for $i = 0, 1$. However we cannot afford to compute the vertices of the levels explicitly if we are aiming for an $O(n \log n)$-time algorithm, as the best known upper bound on the complexity of a level in $\mathcal{A}(H_i)$ is $O(n^{4/3})$ [14], and a lower bound of $n \cdot 2^{\Omega(\sqrt{\log n})}$ exists [30]. We use Chan's randomized technique for solving LP-type problems in which the constraints are defined implicitly by a set of input objects, and which satisfy certain properties (see Lemma 3.1 below).

Given a set $\mathbb{H}$ of constraints and a totally ordered set W, a weight function $\omega : 2^{\mathbb{H}} \to W$ is called *LP-type* of dimension at most d if the following three conditions are satisfied for every subset $H \subseteq \mathbb{H}$ and each constraint $h \in \mathbb{H}$:

- There exists a subset B of size at most d, called a *basis* of H, so that $\omega(H) = \omega(B)$.
- $\omega(H \cup \{h\}) \geq \omega(H)$.
- Let $F \subseteq H$ such that $\omega(F) = \omega(H)$. Then $\omega(H \cup \{h\}) > \omega(H) \Leftrightarrow \omega(F \cup \{h\}) > \omega(F)$.

Since linear programming, with ω being the corresponding linear objective function, is an LP-type problem of dimension $d + 1$, (1) is an LP-type problem. See [29] for more details. The following lemma is the main result behind Chan's technique.

Lemma 3.1 (Chan [7]) *Let $\omega : 2^{\mathbb{H}} \to W$ be an LP-type function of constant dimension d, and let $\alpha < 1$ and s be fixed constants. Suppose $f : \mathbb{P} \to 2^{\mathbb{H}}$ is a function that maps inputs from some set $\mathbb{P}$ to sets of constraints with the following properties:*

(C1) For inputs $P_1, \ldots, P_d \in \mathbb{P}$ of constant size, a basis for $f(P_1) \cup \cdots \cup f(P_d)$ can be computed in constant time.

(C2) For any input $P \in \mathbb{P}$ of size n and any basis $B \subseteq f(P)$, we can decide in $O(D(n))$ time whether B satisfies $f(P)$, i.e., $\omega(f(P)) = \omega(B)$.

(C3) For any input $P \in \mathbb{P}$ of size n, we can construct, in $O(D(n))$ time, inputs $P_1, \ldots, P_s \in \mathbb{P}$ each of size at most $\lceil \alpha n \rceil$, so that $f(P) = f(P_1) \cup \cdots \cup f(P_s)$.

Then we can compute a basis for $f(P)$ in $O(D(n))$ expected time, assuming that $D(n)/n^{\varepsilon}$ is monotonically increasing.

This lemma is a multidimensional version of an earlier technique that Chan proposed in [6]; he used this technique to compute the Tukey depth of a point set. A very slight (straightforward) variant of this algorithm can be used to solve our problem. For the sake of completeness, we sketch the algorithm here.

We formulate the problem in a slightly more general framework. The set of inputs $\mathbb{P}$ that we consider is the set of tuples $(G_0, G_1, \tau, a_0, a_1, b_0, b_1)$, where $G_i \subseteq H_i$, for $i = 0, 1$, τ is a triangle in the plane, and a_0, a_1, b_0, b_1 are nonnegative integers. The function $f : \mathbb{P} \to 2^{\mathbb{R}^2}$ is given by

$$f(G_0, G_1, \tau, a_0, a_1, b_0, b_1) = \tau \cap (\mathcal{L}_{a_0}(G_0) \cup \mathcal{L}_{a_1}(G_1) \cup \mathcal{U}_{b_0}(G_0) \cup \mathcal{U}_{b_1}(G_1)). \tag{2}$$

The points of $L = (\mathcal{L}_{a_0}(G_0) \cup \mathcal{L}_{a_1}(G_1)) \cap \tau$ are colored red, and the points of $U = (\mathcal{U}_{b_0}(G_0) \cup \mathcal{U}_{b_1}(G_1)) \cap \tau$ are colored blue. We wish to compute $\omega(f(G_0, G_1, \tau, a_0, a_1, b_0, b_1))$, as defined in (1).

We show that (2) satisfies (C1)–(C3). Condition (C1) is trivial because we can solve the problem explicitly in $O(1)$ time for constant-size inputs, by constructing the full arrangement of the input lines.

As for (C2), let ℓ be the line defined by a basis B. We need to determine whether ℓ lies above L and below U. We describe how to determine whether ℓ lies above L. Let τ^+ be the portion of τ lying above ℓ, then ℓ lies above L if and only if $\tau^+ \cap L = \emptyset$; the latter holds if and only if none of the edges of τ^+ intersects L. Let e be an edge of τ^+. We compute the intersection points of e with the lines in $G_0 \cup G_1$ and sort them along e. By computing the level of an endpoint of e with respect to G_0 and G_1 and then traversing the list of the intersection points, we can determine in linear time whether L intersects e. Hence (C2) holds with $D(n) = O(n \log n)$.

As for (C3), we choose a constant r and compute in linear-time a $(1/r)$-cutting Ξ of $G_0 \cup G_1$ of size $O(r^2)$ within τ [10]. For a triangle $\Delta \in \Xi$, let $G_i^{\Delta} \subseteq G_i$ be the set of lines that intersect Δ. Let a_i^{Δ} (resp., b_i^{Δ}) be the number of lines of G_i that lie below (resp., above) Δ. Then

$$\begin{aligned} f(G_0, G_1, \tau, a_0, a_1, b_0, b_1) &= \bigcup_{\Delta \in \Xi} f(G_0, G_1, \Delta, a_0, a_1, b_0, b_1) \\ &= \bigcup_{\Delta \in \Xi} f(G_0^{\Delta}, G_1^{\Delta}, \Delta, a_0 - a_0^{\Delta}, a_1 - a_1^{\Delta}, b_0 - b_0^{\Delta}, b_1 - b_1^{\Delta}). \end{aligned}$$

Since $|G_0^\Delta \cup G_1^\Delta| \leq |G_0 \cup G_1|/r$, condition (C3) is satisfied.

Hence, we can compute a basis for $f(H_0, H_1, \mathbb{R}^2, k_0, k_1, k_0, k_1)$ in randomized expected $O(n \log n)$ time. Putting everything together, we conclude the following.

Theorem 3.2 *Given two finite point sets P_0, P_1 in $\mathbb{R}^3$ with a total of n points and a direction u, we can compute a center-transversal line for P_0, P_1 in direction u, or decide that no such line exists, in $O(n \log n)$ expected time.*

4 Variations

Bichromatically deepest line. The algorithm that we have presented in Section 2 can be extended so that, for any given number $\alpha \in [0,1]$, it finds a line ℓ with the property that any closed halfspace containing ℓ also contains at least $\lceil \alpha |P_i| \rceil$ points of P_i, for $i = 0, 1$, or determines that no such line exists. The running time remains $O(n^{1+\varepsilon}\kappa^2(n))$, for any $\varepsilon > 0$.

We define the *bichromatic depth* of a line ℓ with respect to P_0, P_1 as follows:

$$\textsc{Depth}(\ell; P_0, P_1) = \min_h \left\{ \min \left\{ \frac{|P_0 \cap h|}{|P_0|}, \frac{|P_1 \cap h|}{|P_1|} \right\} \right\} \in [0,1],$$

where the minimum is taken over all closed halfspaces h containing ℓ. Equivalently, $\textsc{Depth}(\ell; P_0, P_1) \geq \alpha$ means that any closed halfspace containing ℓ also contains at least $\lceil \alpha |P_i| \rceil$ points of P_i, for $i = 0, 1$. A line ℓ_0 is a *bichromatically deepest line* if it has maximum bichromatic depth. The center-transversal theorem (Theorem 1.1) implies that there always exists a line of depth at least $1/3$. By conducting a binary search and using the extended version of the algorithm of Section 2, we can easily find a line with maximum depth. We thus obtain the following.

Theorem 4.1 *Given two finite point sets P_0, P_1 in $\mathbb{R}^3$ with a total of n points, we can compute a bichromatically deepest line for P_0, P_1 in $O(n^{1+\varepsilon}\kappa^2(n))$ time, for any $\varepsilon > 0$.*

Computing an almost-deepest line. We next observe that, for any fixed $\delta > 0$, we can compute in linear time a line ℓ whose bichromatic depth with respect to P_0, P_1 is at least $1 - \delta$ times the maximum depth of a line. An *ε-approximation* of a point set P (with respect to closed halfspace ranges) is a subset $A \subseteq P$ such that, for any closed halfspace h we have

$$\left| \frac{|A \cap h|}{|A|} - \frac{|P \cap h|}{|P|} \right| \leq \varepsilon.$$

As is well known [10], for any fixed ε, an ε-approximation of size $O\left(\frac{1}{\varepsilon^2} \log \frac{1}{\varepsilon}\right)$ can be computed deterministically in $O(n)$ time.

We fix $\varepsilon = \frac{\delta}{6}$, and compute for each P_i an ε-approximation subset $A_i \subset P_i$ as above. We then compute a bichromatic deepest line ℓ_A for A_0 and A_1 in $O(1)$ time and return ℓ_A. We now argue that ℓ_A is an almost-deepest line. Observe that for any line ℓ we have (where h ranges over all closed halfspaces containing ℓ)

$$\begin{aligned}\textsc{Depth}(\ell; P_0, P_1) &= \min_h \min_{i=0,1} \{|P_i \cap h|/|P_i|\} \geq \min_h \min_{i=0,1} \{|A_i \cap h|/|A_i|\} - \varepsilon \\ &= \textsc{Depth}(\ell; A_0, A_1) - \varepsilon,\end{aligned}$$

and similarly

$$\textsc{Depth}(\ell; P_0, P_1) \leq \textsc{Depth}(\ell; A_0, A_1) + \varepsilon.$$

Let ℓ_{opt} be a bichromatically deepest line for P_0, P_1. Since $\textsc{Depth}(\ell_{opt}; P_0, P_1) \geq \frac{1}{3}$, we have

$$\begin{aligned}\textsc{Depth}(\ell_A; P_0, P_1) &\geq \textsc{Depth}(\ell_A; A_0, A_1) - \varepsilon \geq \textsc{Depth}(\ell_{opt}; A_0, A_1) - \varepsilon \\ &\geq \textsc{Depth}(\ell_{opt}; P_0, P_1) - \frac{\delta}{3} \geq (1-\delta)\textsc{Depth}(\ell_{opt}; P_0, P_1).\end{aligned}$$

We thus conclude the following.

Theorem 4.2 *For a fixed parameter $\delta > 0$, and two finite point sets $P_0, P_1 \subset \mathbb{R}^3$ with a total of n points, we can compute in $O(n)$ time a line ℓ whose bichromatic depth is at least $1-\delta$ times the maximum bichromatic depth.*

5 Conclusions

The efficiency of our algorithm in Section 2 depends on the worst-case complexity $\kappa(n)$ of a k-level in an arrangement of n planes in three dimensions. The currently best known bound $\kappa(n) = O(n^{5/2})$ of [28] is probably not tight, and reducing it would have direct impact on the running time bound of our algorithm. Also, it is not clear that our approach best exploits the geometric structure of the problem in $\mathbb{R}^3$. For example, in Section 3 we note that a center-transversal line exists in a given direction $\vec{u}$ if and only if the projections of P_0 and P_1 onto a plane orthogonal to $\vec{u}$ have a common center point. Can we find an efficient characterization of "candidate" directions, and then test each of them efficiently? Finally, it is unclear whether the tools that we use can be extended to yield any improvement over the algorithm in [5] for constructing center-transversal flats in higher dimensions.

Acknowledgments

We thank Edgar Ramos for early discussions on this problem and the two referees for their useful comments.

References

[1] P. K. Agarwal. Ray shooting and other applications of spanning trees with low stabbing number. *SIAM J. Comput.* 21 (1992), 540–570.

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

[3] P. K. Agarwal and M. Sharir. Arrangements and their applications. In *Handbook of Computational Geometry* (J.-R. Sack and J. Urrutia, eds.), Elsevier Science Publishers, Amsterdam, 2000, 49–119.

[4] P. K. Agarwal, M. Sharir, and E. Welzl. Algorithms for center and Tverberg points. *Proc. 20th Annu. ACM Sympos. Comput. Geom.*, 2004, 61–67.

[5] M. Bern and D. Eppstein. Multivariate regression depth. *Discrete Comput. Geom.* 28 (2002), 1–17.

[6] T. M. Chan. Geometric applications of a randomized optimization technique. *Discrete Comput. Geom.* 22 (1999), 547–567.

[7] T. M. Chan. An optimal randomized algorithm for maximum Tukey depth. *Proc. 15th Annu. ACM-SIAM Sympos. Discrete Algorithms*, 2004, 430–436.

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

[9] B. Chazelle. Cutting hyperplanes for divide-and-conquer. *Discrete Comput. Geom.* 9 (1993), 145–158.

[10] B. Chazelle. *The Discrepancy Method: Randomness and Complexity*. Cambridge University Press, New York, 2001.

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

[12] K. L. Clarkson, D. Eppstein, G. L. Miller, C. Sturtivant, and S.-H. Teng. Approximating center points with iterated Radon points. *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, 1993, 91–98.

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

[14] T. K. Dey. Improved bounds on planar k-sets and related problems. *Discrete Comput. Geom.* 19 (1998), 373–382.

[15] V. Dol'nikov. A generalization of the sandwich theorem. *Mathematical Notes* 52 (1992), 771–779.

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

[17] M. Hubert and P. Rousseeuw. The catline for deep regression. *J. Multivariate Analysis* 66 (1998), 270–296.

[18] S. Jadhav and A. Mukhopadhyay. Computing a centerpoint of a finite planar set of points in linear time. *Discrete Comput. Geom.* 12 (1994), 291–312.

[19] J. Matoušek. Computing the center of planar point sets. *Computational Geometry: Papers from the DIMACS Special Year* (J. E. Goodman, R. Pollack, and W. Steiger, eds.), American Mathematical Society, Providence, 1991, 221–230.

[20] J. Matoušek. *Lectures on Discrete Geometry*. Springer Verlag, Berlin, 2002.

[21] N. Naor and M. Sharir. Computing a point in the center of a point set in three dimensions. *Proc. 2nd Canad. Conf. Comput. Geom.*, 1990, 10–13.

[22] G. Nivasch. An improved, simple construction of many halving edges. In *Surveys on Discrete and Computational Geometry: Twenty Years Later* (J.E. Goodman, J. Pach and R. Pollack, eds.), American Mathematical Society, Providence, RI, 2008, 299–306.

[23] M. Pellegrini. On lines missing polyhedral sets in 3-space. *Discrete Comput. Geom.* 12 (1994), 203–221.

[24] M. Pellegrini. Ray shooting and lines in space. editors, *Handbook of Discrete and Computational Geometry, 2nd ed.* (J. E. Goodman and J. O'Rourke, eds.), CRC Press LLC, Boca Raton, FL, 2004, 839–856.

[25] M. Pellegrini and P. Shor. Finding stabbing lines in 3-space. *Discrete Comput. Geom.* 8 (1992), 191–208.

[26] P. Rousseeuw and M. Hubert. Depth in an arrangement of hyperplanes. *Discrete Comput. Geom.* 22 (1999), 167–176.

[27] P. Rousseeuw and M. Hubert. Regression depth. *J. Amer. Stat. Assoc.* 94 (1999), 388–402.

[28] M. Sharir, S. Smorodinsky, and G. Tardos. An improved bound for k-sets in three dimensions. *Discrete Comput. Geom.* 26 (2001), 195–204.

[29] M. Sharir and E. Welzl. A combinatorial bound for linear programming and related problems. *Proc. 9th Annu. Sympos. Theoretical Aspects of Computer Science*, 1992, 569–579.

[30] G. Tóth. Point sets with many k-sets. *Discrete Comput. Geom.* 26 (2001), 187–194.

[31] R. T. Živaljević and S. T. Vrećica. An extension of the ham sandwich theorem. *Bull. London Math. Soc.* 22 (1990), 183–186.