STAR UNFOLDING OF A POLYTOPE WITH APPLICATIONS*

PANKAJ K. AGARWAL†, BORIS ARONOV‡, JOSEPH O'ROURKE§, AND CATHERINE A. SCHEVON¶

Abstract. We introduce the notion of a *star unfolding* of the surface $\mathcal{P}$ of a three-dimensional convex polytope with n vertices, and use it to solve several problems related to shortest paths on $\mathcal{P}$.

The first algorithm computes the edge sequences traversed by shortest paths on $\mathcal{P}$ in time $O(n^6\beta(n)\log n)$, where $\beta(n)$ is an extremely slowly-growing function. A much simpler $O(n^6)$ time algorithm that finds a small superset of all such edge sequences is also sketched.

Second, we describe an $O(n^8 \log n)$ time procedure for computing the *geodesic diameter* of $\mathcal{P}$: the maximum possible separation of two points on $\mathcal{P}$, with the distance measured along $\mathcal{P}$.

Finally, we describe an algorithm that preprocesses $\mathcal{P}$ into a data structure that can efficiently answer the queries of the form: "Given two points, what is the length of the shortest path connecting them?" Given a parameter $1 \le m \le n^2$, it can preprocess $\mathcal{P}$ in time $O(n^6 m^{1+\delta})$, for any $\delta > 0$, into a data structure of size $O(n^6 m^{1+\delta})$, so that a query can be answered in time $O((\sqrt{n}/m^{1/4})\log n)$. If one query point always lies on an edge of $\mathcal{P}$, the algorithm can be improved to use $O(n^5 m^{1+\delta})$ preprocessing time and storage and guarantee $O((n/m)^{1/3}\log n)$ query time, for any choice of m between 1 and n.

1. Introduction. The problem of computing shortest paths in Euclidean space amidst polyhedral obstacles arises in planning optimal collision-free paths for a given robot, and has been widely studied. In two dimensions, the problem has been thoroughly explored and a number of efficient algorithms have been developed, see e.g. [SS86, Wel85, Mit93, HS93]. However, the problem becomes significantly harder in three dimensions. Canny and Reif [CR87] have shown it to be NP-hard, and the fastest available algorithm runs in singly-exponential time [RS89, Sha87]. This has motivated researchers to develop efficient approximation algorithms [Pap85, Cla87, CSY94, HS95] and to study interesting special cases [MMP87, Sha87]. One of the most widely studied special cases is computing shortest paths on the surface of a convex polytope [SS86, MMP87, Mou90]; this problem was originally formulated by H. Dudeney in 1903; see [Gar61, p. 36]. Sharir and Schorr presented an $O(n^3 \log n)$ algorithm for this problem, which was subsequently improved by Mitchell et al. [MMP87] to $O(n^2 \log n)$, and then by Chen and Han to $O(n^2)$ [CH90].

In this paper we consider three problems involving shortest paths on the surface $\mathcal{P}$ of a convex polytope in $\mathbb{R}^3$. A shortest path on $\mathcal{P}$ is identified uniquely by its endpoints and the sequence of edges that it encounters. Sharir [Sha87] proved that no more than $O(n^7)$ distinct sequences of edges are actually traversed by the shortest paths on $\mathcal{P}$. This bound was subsequently improved to $\Theta(n^4)$ [Mou85, SO88]. Sharir also gave an $O(n^8 \log n)$ time algorithm to compute an $O(n^7)$ size superset of shortest-path edge sequences. However computing the exact set of shortest-path edge sequences

* An earlier version of this paper was presented at the Second Scandinavian Workshop on Algorithm Theory [AAOS90]. Part of the work was carried out when the first two authors were at Courant Institute of Mathematical Sciences, New York University and later at DIMACS (Center for Discrete Mathematics and Theoretical Computer Science), a National Science Foundation Science and Technology Center — NSF-STC88-09648, and the fourth author was at the Department of Computer Science, The Johns Hopkins University. The work of the first author is supported by National Science Foundation Grant CCR-91-06514. The work of the second author was also partially supported by an AT&T Bell Laboratories Ph.D. Scholarship and NSF Grant CCR-92-11541. The third author's work is supported by NSF grants CCR-88-2194 and CCR-91-22169.

† Computer Science Department, Duke University, Durham, NC 27706 USA

‡ Computer Science Department, Polytechnic University, Brooklyn, NY 11201 USA

§ Department of Computer Science, Smith College, Northampton, MA 01063 USA

¶ AT&T Bell Laboratories, P.O. Box 636, Murray Hill, NJ 07974 USA

seems to be very difficult. Schevon and O'Rourke [SO89] presented an algorithm that computes the exact set of all shortest-path edge sequences and also identifies, in logarithmic time, the edge sequences traversed by all shortest paths connecting a given pair of query points lying on edges of $\mathcal{P}$. The sequences can be explicitly generated, if necessary, in time proportional to their length. Their algorithm, however, requires $O(n^9 \log n)$ time and $O(n^8)$ space.[1]

In this paper we propose two edge-sequence algorithms. The first is a simple $O(n^6)$ algorithm to compute a superset of shortest-path edge sequences, thus improving the result of [Sha87]; it is described in Section 5. The second computes the exact set of shortest-path edge sequences in $O(n^6 \beta(n) \log n)$ time, where $\beta(\cdot)$ is an extremely slowly-growing function, a significant improvement over the previously mentioned $O(n^9 \log n)$ algorithm. The computation of the collection of all shortest-path edge sequences on a polytope is an intermediate step of several algorithms [Sha87, OS89], and is of interest in its own right.

The second problem studied in this paper is that of computing the *geodesic diameter* of $\mathcal{P}$, i.e., the maximum distance along $\mathcal{P}$ between any two points on $\mathcal{P}$. O'Rourke and Schevon [OS89] gave an $O(n^{14} \log n)$ time procedure for determining the geodesic diameter of $\mathcal{P}$. In [AAOS90], we presented a simpler and faster algorithm whose running time is $O(n^{10})$. We improve this to $O(n^8 \log n)$ here.

The third problem involves answering queries of the form: "Given $x, y \in \mathcal{P}$, determine the distance between x and y along $\mathcal{P}$." Given a parameter $1 \le m \le n^2$, we present a method for preprocessing $\mathcal{P}$, in $O(n^6 m^{1+\delta})$ time, into a data structure of size $O(n^6 m^{1+\delta})$ for any $\delta > 0$, so that a query can be answered in time $O((\sqrt{n}/m^{1/4}) \log n)$. If x is known to lie on an edge of the polytope, the preprocessing and storage requirements are reduced to $O(n^5 m^{1+\delta})$ and the query time becomes $O((n/m)^{1/3} \log n)$ for $1 \le m \le n$. Constants of proportionality in the above bounds depend of the choice of δ.

Our algorithms are based on a common geometric concept, the *star unfolding*. Let $x \in \mathcal{P}$ be a point such that the shortest path from x to every vertex of $\mathcal{P}$ is unique. Intuitively, the star unfolding of $\mathcal{P}$ with respect to this point is obtained by removing these n shortest paths from $\mathcal{P}$ and embedding the remaining surface isometrically in the plane. Remarkably, the star unfolding is isometric to a simple planar polygon and the structure of shortest paths emanating from x on $\mathcal{P}$ corresponds to a certain Voronoi diagram in the plane [AO92]. Together with relative stability of the combinatorial structure of the unfolding as x moves within a small neighborhood on $\mathcal{P}$, these properties facilitate the construction of efficient algorithms for the above three problems. Although all the algorithms have high polynomial time complexity as a function of n, they are not so inefficient in relation to the worst-case number of edge sequences, $\Theta(n^4)$.

Chen and Han [CH90] have independently discovered the star unfolding and used it for computing the shortest-path information from a single fixed point on the surface of a polytope. The nonoverlap of the unfolding [AO92], however, was not known at the time of their work.

This paper is organized as follows. In Section 2, we formalize our terminology and list some basic properties of shortest paths. Section 3 defines the star unfolding and

[1] A preliminary version of this algorithm [SO88] erroneously claimed a time complexity of $O(n^7 2^{\alpha(n)} \log n)$; this claim was corrected in [SO89]. Hwang et al. [HCT89] suggested a more efficient procedure for solving the same problem, but the key claim (their second Lemma 6) has yet to be convincingly established [Sch89, Chapter 5].

establishes some of its properties. Section 4 sketches an efficient algorithm to compute a superset of all possible shortest-path edge sequences, and in Section 5 we present an algorithm for computing the exact set of these sequences; both algorithms are based on the star unfolding. In Section 6 we again use the notion of star unfolding to obtain a faster algorithm for determining the geodesic diameter of a convex polytope. Section 7 deals with shortest-path queries. Section 8 contains some concluding remarks and open problems.

2. Geometric Preliminaries. We begin by reviewing the geometry of shortest paths on convex polytopes.

Let $\mathcal{P}$ be the surface of a polytope with n vertices. We refer to vertices of $\mathcal{P}$ as *corners*; the unqualified terms *face* and *edge* are reserved for faces and edges of $\mathcal{P}$. We assume that $\mathcal{P}$ is triangulated. This does not change the number of faces and edges of $\mathcal{P}$ by more than a multiplicative constant, but simplifies the description of our algorithms.

2.1. Geodesics and Shortest Paths. A path π on $\mathcal{P}$ that cannot be shortened by a local change at any point in its relative interior is referred to as a *geodesic*. Equivalently, a geodesic on the surface of a convex polytope is either a subsegment of an edge or a path that (1) does not pass through corners, though it may possibly terminate at them, (2) is straight near any point in the interior of a face and (3) is transverse to every edge it meets in such a fashion that it would appear straight if one were to "unfold" the two faces incident on this edge until they lie in a common plane; see, for example, Sharir and Schorr [SS86]. The behavior of a geodesic is thus fully determined by its starting point and initial direction. In the following discussion we disregard the geodesics lying completely within a single edge of $\mathcal{P}$. Given the sequence of edges a geodesic crosses and its starting and ending points, the geodesic itself can be obtained by laying the faces that it visits out in the plane, in order, so that adjacent faces share an edge and lie on opposite sides of it, and then connecting the (images of) the two endpoints by a straight-line segment. In particular, the sequence of traversed edges together with the endpoints completely determine the geodesic.

Trivially every shortest path along $\mathcal{P}$ is a geodesic and no shortest path meets a face or an edge more than once. We call the length of a shortest path between two points $p, q \in \mathcal{P}$ the *geodesic distance* between p and q, and denote it by $d(p,q)$. The following additional properties of shortest paths are crucial for our analysis.

LEMMA 2.1 (SHARIR AND SCHORR [SS86]). *Let π_1 and π_2 be distinct shortest paths emanating from x. Let $y \in \pi_1 \cap \pi_2$ be a point distinct from x. Then either one of the paths is a subpath of the other, or neither π_1 nor π_2 can be extended past y while remaining a shortest path.*

COROLLARY 2.2. *Two shortest paths cross at most once.*

LEMMA 2.3. *If π_1, π_2 are two distinct shortest paths connecting $x, y \in \mathcal{P}$, each of the two connected components of $\mathcal{P} \setminus (\pi_1 \cup \pi_2)$ contains a corner.*

Proof. First, Lemma 2.1 implies that removal of $\pi_1 \cup \pi_2$ splits $\mathcal{P}$ into exactly two components. If one of the two components of $\mathcal{P} \setminus (\pi_1 \cup \pi_2)$ contained no corners, π_1 and π_2 would have to traverse the same sequence of edges and faces. However, there exists at most one geodesic connecting a given pair of points and traversing a given sequence of edges and faces. □

2.2. Edge Sequences and Sequence Trees. A *shortest-path edge sequence* is the sequence of edges intersected by some shortest path π connecting two points on $\mathcal{P}$, in the order met by π. Such a sequence is *maximal* if it cannot be extended in either direction while remaining a shortest-path edge sequence; it is *half-maximal* if no extension is possible at one of the two ends. It has been shown by Schevon and O'Rourke [SO88] that the maximum total number of half-maximal sequences is $\Theta(n^3)$.

Observe that every shortest-path edge sequence σ is a prefix of some half-maximal sequence, namely the one obtained by extending σ maximally at one end. Thus an exhaustive list of $O(n^3)$ half-maximal sequences contains, in a sense, all the shortest-path edge-sequence information of $\mathcal{P}$. More formally, given an arbitrary collection of edge sequences emanating from a fixed edge e, let the *sequence tree* Σ of this set be the tree with all distinct non-empty prefixes of the given sequences as nodes, the trivial sequence consisting solely of e as the root, and such that σ is an ancestor of σ' in the tree if and only if σ is a prefix of σ' [HCT89]. The $\Theta(n^3)$ bound on the number of half-maximal sequences implies that the collection of $O(n)$ sequence trees obtained by considering all shortest-path edge sequences from each edge of $\mathcal{P}$ in turn, has a total of $\Theta(n^3)$ leaves and $\Theta(n^4)$ nodes in the worst case.

2.3. Ridge Trees and the Source Unfolding. The shortest paths emanating from a fixed source $x \in \mathcal{P}$ cover the surface of $\mathcal{P}$ in a way that can be naturally represented by "unfolding" the paths to a planar layout with respect to x. This unfolding, the "source unfolding," has been studied since the turn of the century. We will define it precisely in a moment. A second way to organize the paths in the plane is the "star unfolding," to be defined in Section 3. This is not quite as natural, and of more recent lineage. Our algorithms will be built around the star unfolding, but some of the arguments do refer to the source unfolding as well.

Given two points x, y on $\mathcal{P}$, $y \in \mathcal{P}$ is a *ridge point* with respect to x if there is more than one shortest path between x and y. Ridge points with respect to x form a *ridge tree* T_x embedded on $\mathcal{P}$,[2] whose leaves are corners of $\mathcal{P}$, and whose internal vertices have degree at least three and correspond to points of $\mathcal{P}$ with three or more distinct shortest paths to x. In a degenerate situation where x lies on the ridge tree for some corner p, then p will not be a leaf of T_x, but rather will become a degree-2 vertex in T_x; so in general not all corners will appear as leaves of T_x. We define a *ridge* as a maximal subset of T_x consisting of points with exactly two distinct shortest paths to x, and containing no corners of $\mathcal{P}$. These are the "edges" of T_x. Ridges are open geodesics [SS86]; a stronger characterization of ridges is given in Lemma 2.4. Fig. 1 shows two views of a ridge tree on a pyramid.

We will refer to a point $y \in \mathcal{P}$ as a *generic* point if it is not a ridge point with respect to any corner of $\mathcal{P}$. The maximal connected portion of a face (resp. an edge) of $\mathcal{P}$ consisting entirely of generic points will be called a *ridge-free region* (resp. an *edgelet*). See Fig. 2.

If we cut $\mathcal{P}$ along the ridge tree T_x and isometrically embed the resulting set in $\mathbb{R}^2$, we obtain the *source unfolding* of [OS89].[3] In the source unfolding, the ridges lie on the boundary of the unfolding, while x lies at its "center," which results in a star-shaped polygon [SS86]; see Fig. 3. Let a *peel* be the closure of a connected component of the set obtained by removing from $\mathcal{P}$ both the ridge tree T_x and the shortest paths from x to all corners. A peel is isometric to a convex polygon [SS86]. Each peel's boundary consists of x, the shortest paths to two "consecutive" corners of $\mathcal{P}$, p and p', and the unique path in T_x connecting p to p'. A peel can be thought

[2] For smooth surfaces (Riemannian manifolds), the ridge tree is known as the "cut locus" [Kob67].

[3] The same object is called $U(\mathcal{P})$ in [SS86], "planar layout" in [Mou85], and "outward layout" in [CH90]. For Riemannian manifolds, it is the "exponential map" [Kob67].

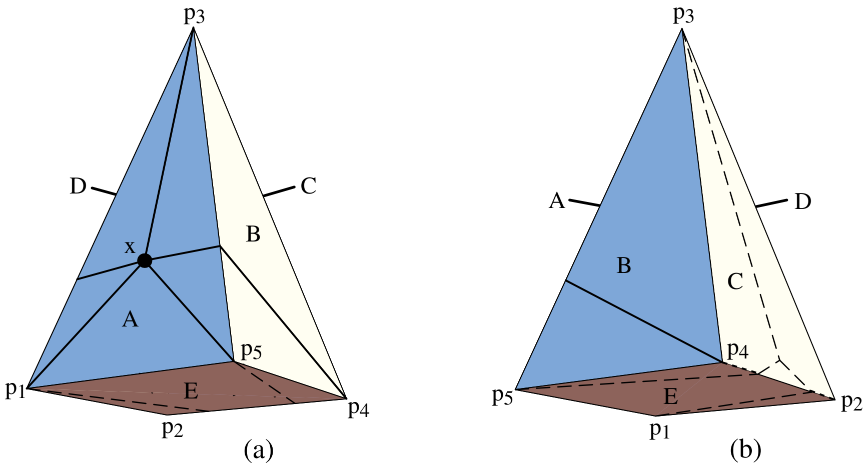

FIG. 1. *Pyramid, front (a) and side (b) view. Shortest paths to five vertices from source x are shown solid; the ridge tree is dashed. Coordinates of vertices are $(\pm 1, \pm 1, 0)$ for p_1, p_2, p_4, p_5, and $p_3 = (0, 0, 4)$; $x = (0, 3/4, 1)$. The ridges incident to p_2 and p_4 lie nearly on the edge p_2p_4.*

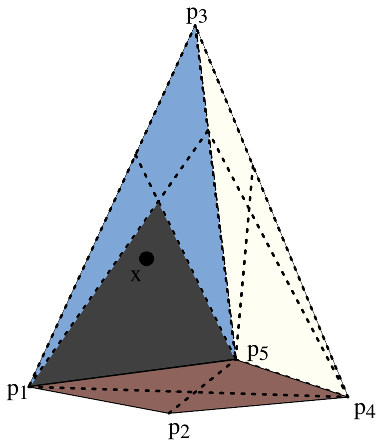

FIG. 2. *Ridge-free regions for Fig. 1. $T_{p_1} \cup \cdots \cup T_{p_5}$ are shown dashed (e.g., T_{p_3} is the 'X' on the bottom face). The ridge-free region containing x is shaded darker.*

of as the collection of all the shortest paths emanating from x between xp and xp'. (The peel between xp_1 and xp_5 is shaded in Fig. 3.)

We need to strengthen the characterization of ridges from geodesics to shortest paths, in order to exploit Corollary 2.2. This characterization seems to be new.

LEMMA 2.4. *Every ridge of the ridge tree T_x, for any point $x \in \mathcal{P}$, is a shortest path.*

Proof. An edge π of the ridge tree is an (open) geodesic consisting of points that have two different shortest paths to x [SS86]. Suppose π is not a shortest path. Then since it is composed of segments, it contains shortest paths, and in particular, a shortest path π' delimited by two points $a, b \in \pi$ such that there is another shortest path π'' connecting them. Refer to Figure 4. By Lemma 2.1, $\pi' \cap \pi'' = \{a, b\}$. Let α_1 and α_2 be the two shortest paths from x to a, and β_1 and β_2 be the two shortest

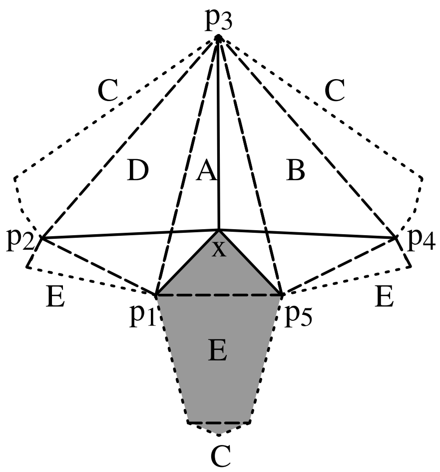

FIG. 3. *Source unfolding for the example in Fig. 1. Shortest paths to vertices are solid, polytope edges are dashed, and ridges are dotted. One peel is shaded.*

paths from x to b. Notice that π', α_1, α_2, β_1, β_2, do not meet except at the endpoints, by Lemma 2.1. In particular, we can relabel these paths so that α_2 and β_2 lie in the same connected component of $\mathcal{P} - (\alpha_1 \cup \beta_1 \cup \pi')$. There are two cases to consider:

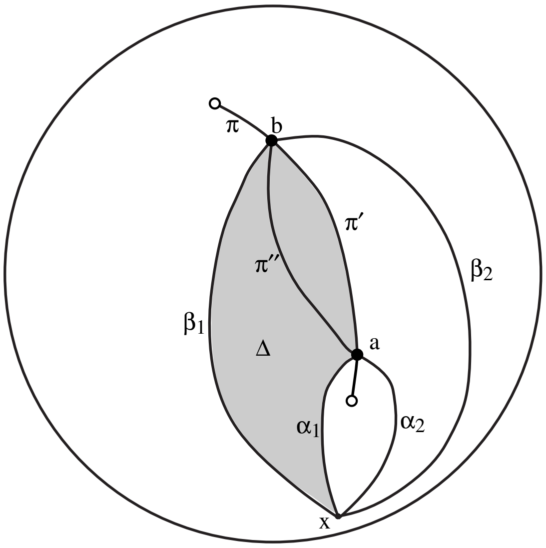

FIG. 4. *Illustration of the proof of Lemma 2.4. Here x is the source, and π a geodesic ridge, with $\pi' \subseteq \pi$ a shortest path. The region Δ cannot contain any vertices of $\mathcal{P}$.*

Case 1: $x \notin \pi''$. Thus π'' does not meet α_1 or α_2 except at a, by Lemma 2.1. Similarly, π'' does not meet β_1 or β_2 except at b. Thus, without loss of generality, we can assume that π'' lies in the portion Δ of $\mathcal{P}$ bounded by π', α_1, and β_1, and not containing α_2 or β_2. Since α_1, β_1 are shortest paths from x to a and b, respectively, their relative interiors do not intersect T_x. Moreover, π' does not contain a vertex of T_x, so Δ does not contain any corner of $\mathcal{P}$, as each corner is a vertex of T_x. On the other hand, paths $\pi', \pi'' \subset \Delta$ are distinct shortest paths connecting a to b, so by Lemma 2.3 each of the two connected components of $\mathcal{P} \setminus (\pi' \cup \pi'')$ has to contain a corner of $\mathcal{P}$. However, one of these components is entirely contained in Δ—a contradiction.

Case 2: $x \in \pi''$. As π'' and α_1 can be viewed as emanating from a and having x in common, and π'' extends past x, Lemma 2.1 implies that α_1 is a prefix of π''. Similarly, α_2 is a prefix of π'', contradicting distinctness of α_1 and α_2.

□

Remark. Case 2 in the above proof is vacuous if x is a corner, which is the case in our applications of this lemma.

As defined, T_x is a tree with n vertices of degree less than 3 and thus has $\Theta(n)$ vertices and edges. However, the worst-case combinatorial size of T_x jumps from $\Theta(n)$ to $\Theta(n^2)$ if one takes into account the fact that a ridge is a shortest path comprised of as many as $\Theta(n)$ line segments on $\mathcal{P}$ in the worst case—and it is possible to exhibit a ridge tree for which the number of ridge-edge incidences is indeed $\Omega(n^2)$ [Mou85]. For simplicity we assume that ridges intersect each edge of $\mathcal{P}$ transversely.

3. Star Unfolding. In this section we introduce the notion of the *star unfolding* of $\mathcal{P}$ and describe its geometric and combinatorial properties. Working independently, both Chen and Han [CH90, CH91] and Rasch [Ras90] have used the same notion, and in fact the idea can be found in Aleksandrov's work [Ale58, p.226] [AZ67, p.171].

3.1. Geometry of the Star Unfolding. Let $x \in \mathcal{P}$ be a generic point, so that there is a unique shortest path connecting x to each corner of $\mathcal{P}$. These paths are called *cuts* and are comprised of *cut points* (see Fig. 1). If $\mathcal{P}$ is cut open along these cuts the result is a two-dimensional complex that we call the *star unfolding* S_x. If isometrically embedded in the plane, the star unfolding corresponds to a simple polygon. That the star unfolding can be embedded in the plane without overlap is by no means a straightforward claim; it was first established in [AO92]:

Lemma 3.1 (Aronov and O'Rourke [AO92]). *If viewed as a metric space with the natural definition of interior metric, S_x is isometric to a simple polygon in the plane (with the internal geodesic metric).* The polygonal boundary ∂S_x consists entirely of edges originating from cuts. The vertices of S_x derive from the corners of $\mathcal{P}$ and from the source x. An example is shown in Fig. 5. More complex examples will be shown in Fig. 7.

The cuts partition the faces of $\mathcal{P}$ into subfaces, which map to what we call the *plates* of S_x. Since we assume the faces of $\mathcal{P}$ to be triangles, each plate is a compact convex polygon with at most six edges. See Fig. 6(b). We consider these plates to be the faces of the two-dimensional complex S_x. We assume that the complex carries with it labeling information consistent with $\mathcal{P}$.

Somewhat abusing the notation, we will freely switch between viewing S_x as a complex and as a simple polygon embedded in the plane. In particular, a path $\pi \subset S_x$ will be referred to as a *segment* if it corresponds to a straight-line segment in this embedding. Note that every segment in S_x is a shortest path in the intrinsic metric of the complex, but not every shortest path in S_x is a segment, as some shortest paths in S_x might bend at corners.

For $p \in \mathcal{P}$, let $U(p)$ be the set of points in S_x to which p maps; U is the "unfolding" map (with respect to x). $U(p)$ is a single point whenever p is not a cut point. A non-corner point $y \in \mathcal{P}$ distinct from x and lying on a cut has exactly two unfolded images in S_x. The corners of $\mathcal{P}$ map to single points. $X = U(x) = \{x_1, \ldots, x_n\}$ is a set of n

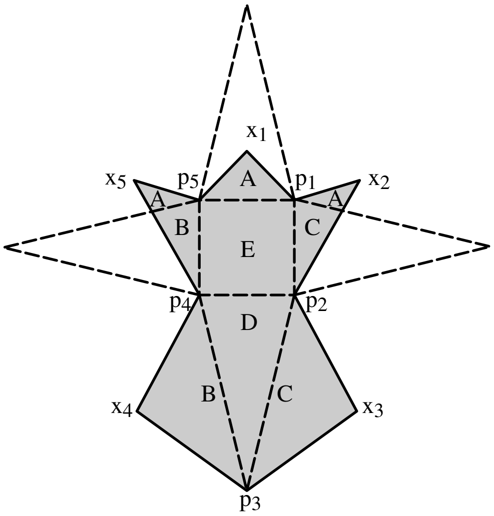

FIG. 5. *Construction of the star unfolding corresponding to Fig. 1. S_x is shaded. The superimposed dashed edges show the "natural" unfolding obtained by cutting along the four edges incident to p_3. The A, B, C, D, and E labels indicate portions of S_x derived from those faces.*

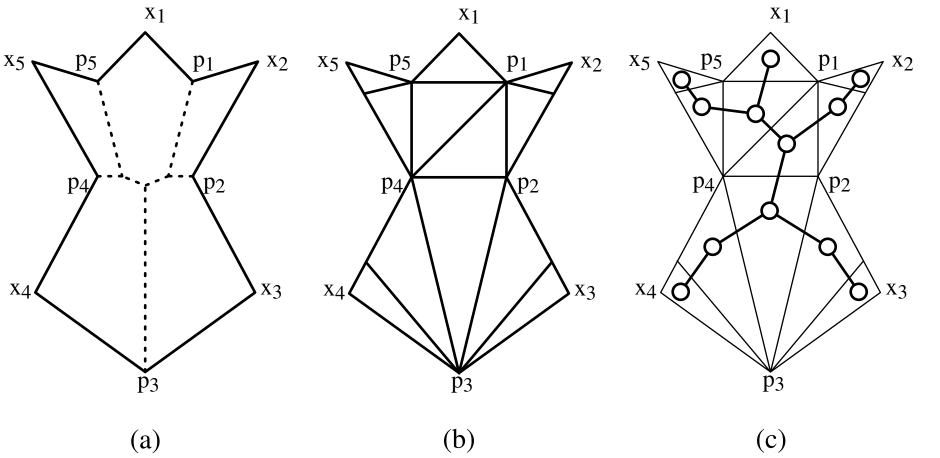

FIG. 6. *(a) Ridge tree, (b) plates, and (c) pasting tree corresponding to Fig. 5.*

distinct points in S_x. If $|U(p)| = 1$, then, with a slight abuse of notation, we use $U(p)$ to denote the unique image of p as well. We can extend the definition of the map U to sets in a natural way by putting $U(Q) = \bigcup_{q \in Q} U(q)$. In particular, we have:

LEMMA 3.2 (SHARIR AND SCHORR [SS86]). *For a point $y \in \mathcal{P}$, any shortest path π from x to y maps to a segment $\pi^* \subset S_x$ connecting an element of $U(y)$ to an element of $U(x)$.*

There is also a view of S_x that relates it to the source unfolding: the star unfolding is just an "inside-out" version of the source unfolding, in the following sense. The star unfolding can be obtained by stitching peels together along ridges; see Fig. 6(a). The source unfolding is obtained by gluing them along the cuts; compare with Fig. 3. (A "peel" was defined in Section 2.3 as a subset of $\mathcal{P}$, but slightly abusing the terminology we also use this term to refer to the corresponding set of points in the source or star unfolding.)

We next define the *pasting tree* Π_x as the graph whose nodes are the plates of S_x, with two nodes connected by an arc if the corresponding plates share an edge in S_x. See Fig. 6(c). For a generic point x, Π_x is a tree with $O(n^2)$ nodes, as it is the dual of a convex partition of a simple polygon without Steiner points. (If x were a ridge point of some corner, S_x would not be connected and Π_x would be a forest.) Π_x has only n leaves, corresponding to the triangular plates incident to the n images of x in S_x. By Lemma 3.2, any shortest path from x to $y \in \mathcal{P}$ corresponds to a simple path in Π_x, originating at one of the leaves. Thus, the $O(n^3)$ edge sequences corresponding to the simple paths that originate from leaves of Π_x include all shortest path edge sequences emanating from x. In fact, there are $O(n^2)$ maximal edge sequences in this set, one for each pair of leaves. This relation between Π_x and the shortest path sequences is crucial in our sequence algorithms, described in Sections 4 and 5.

In the following sections we will need the concept of the "kernel" of a star unfolding. Number the corners $p_1, \dots, p_n$ in the order in which cuts emanate from x. Number the n source images (elements of $X = U(x)$) so that ∂S_x is the cycle $x_1 p_1 x_2 \dots p_n x_1$ comprised of $2n$ segments (see Fig. 5). The kernel is a subset of S_x; here it is more convenient to view S_x as a simple polygon. Consider the polygonal cycle $p_1 p_2 \dots p_n p_1$. We claim that it is the boundary of a simple polygon fully contained in S_x. Indeed, each line segment $p_i p_{i+1}$[4] is fully contained in the peel sandwiched between $x_{i+1} p_i$ and $x_{i+1} p_{i+1}$. Thus the line segments $p_i p_{i+1}$ are segments in S_x in the sense defined above and indeed form a simple cycle. The simple n-gon bounded by this cycle is referred to as the *kernel* K_x of the star unfolding S_x. An equivalent way of defining K_x is by removing from S_x all triangles $\triangle p_{i-1} x_i p_i$, for $i = 1, \dots, n$. As with S_x, we will alternate between viewing K_x as a complex and as a simple polygon in the plane. Fig. 7 illustrates the star unfolding, and its kernel for several randomly generated polytopes.[5] Note that neither set is necessarily star-shaped.

The main property of the kernel that we will need later is:

LEMMA 3.3. *The image of the ridge tree is completely contained within the kernel, which is itself a subset of the star unfolding:* $U(T_x) \subset K_x \subset S_x$.

Proof. Since K_x can be defined by subtraction from S_x, $K_x \subset S_x$ is immediate. The ridge tree T_x can be thought of as the union of the peel boundaries that do not come from cuts. As peels are convex, these boundaries remained when we removed triangles $\triangle p_{i-1} x_i p_i$ from S_x to form K_x. □

Aronov and O'Rourke [AO92] proved that

THEOREM 3.4. $U(T_x)$ *is exactly the restriction of the planar Voronoi diagram of the set* $X = U(x)$ *of source images to within* K_x *or, equivalently, to within* S_x.

3.2. Structure of the Star Unfolding. We now describe the combinatorial structure of S_x. A *vertex* of S_x is an image of x, of a corner of $\mathcal{P}$, or of an intersection of an edge of $\mathcal{P}$ with a cut. An *edge* of S_x is a maximal portion of an image of a cut or an edge of $\mathcal{P}$ delimited by vertices of S_x. It is easy to see that S_x consists of $\Theta(n^2)$ plates in the worst case, even though its boundary is formed by only $2n$ segments, the images of the cuts. We define the *combinatorial structure* of S_x as the 1-skeleton of S_x, i.e., the graph whose nodes and arcs are the vertices and edges of S_x, labeled

[4] Here and thereafter $p_{n+1} = p_1$, $p_0 = p_n$, $x_{n+1} = x_1$, $x_0 = x_n$.

[5] The unfoldings were produced with code written by Julie DiBiase and Stacia Wyman of Smith College.

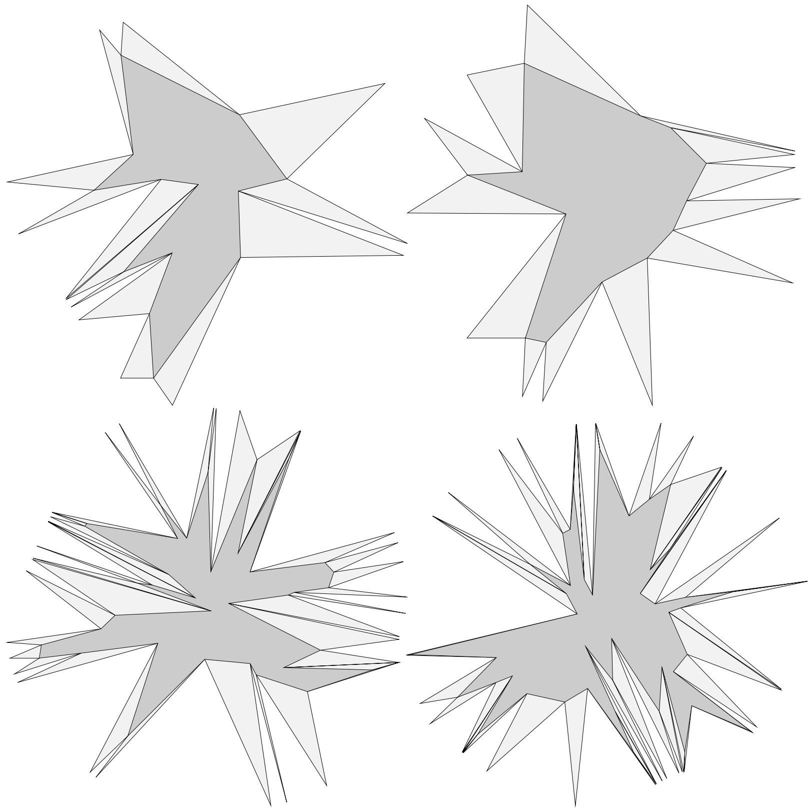

FIG. 7. *Four star unfoldings: $n = 13$, 13, 36, 42 vertices, left to right, top to bottom. The kernel is shaded darker in each figure.*

in correspondence with the labels of S_x, which are in turn derived from labels on $\mathcal{P}$. The combinatorial structure of a star unfolding has the following crucial property:

LEMMA 3.5. *Let x and y be two non-corner points lying in the same ridge-free region or on the same edgelet. Then S_x and S_y have the same combinatorial structure.*

Proof. Let f be the face containing the segment xy in its interior. The case when xy is part of an edge is similar.

As the shortest paths from any point $z \in xy$ to the corners are pairwise disjoint except at z (cf. Lemma 2.1) and z is confined to f, the combinatorial structure of S_z is uniquely determined by

(1) the circular order of the cuts around z, and

(2) the sequence of edges and faces of $\mathcal{P}$ met by each of the cuts.

We will show that (1) and (2) are invariants of S_z as long as $z \in xy$ does not cross a ridge or an edge of $\mathcal{P}$. First, the set of points of f, for which some shortest path to a fixed corner p traverses a fixed edge sequence, is convex—it is simply the intersection of f with the appropriate peel with respect to p—implying invariance of (2).

Now suppose the circular order of the cuts around z is not the same for all $z \in xy$. The initial portions of the cuts, as they emanate from any z, cannot coincide, as distinct cuts are disjoint except at z. Hence there can be a change in this circular order only if one of the vectors pointing along the initial portion of the cuts changes discontinuously at some intermediate point $z' \in xy$. However, this can only happen if z' is a ridge point with respect to a corner, a contradiction. □

This lemma holds under more general conditions. Namely, instead of requiring that xy be free of ridge points, it is sufficient to assume that the number of distinct shortest paths connecting z to any corner does not change as z varies along xy (this number is larger than 1 if xy is a portion of a ridge).

LEMMA 3.6. *Under the assumptions of Lemma 3.5, K_x is isometric to K_y, i.e., they are congruent simple polygons.*

Proof. K_x is determined by the order of corners on $\partial K_x = p_1p_2, \ldots, p_np_1$ and, for each i, by the choice of the shortest path p_ip_{i+1}, if there are two or more such paths. The ordering is fixed, once combinatorial structure of S_x is determined. The choice of the shortest path connecting p_i to p_{i+1} is determined by the constraint that $\triangle p_{i-1}x_ip_i$ is free of corners. □

Let R be a ridge-free region. By the above lemma, S_x can be embedded in the plane in such a way that the images of the corners of $\mathcal{P}$ are fixed for all $x \in R$, while the images of x in S_x move as x varies in $R \subseteq \mathcal{P}$. This guarantees that $K_y = K_x$ for all $x, y \in R$. This is illustrated in Fig. 8. In what follows, we are going to assume

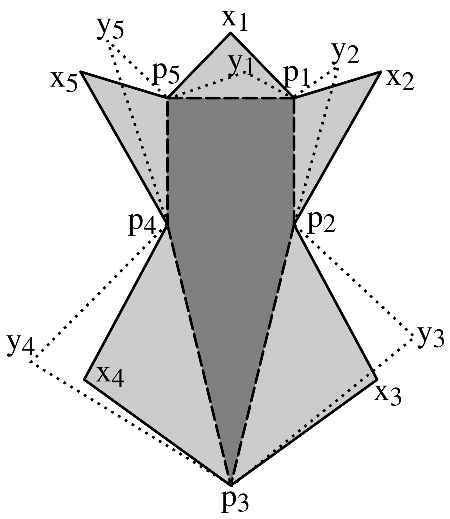

FIG. 8. *The star unfolding when the source x moves to y inside a ridge-free region. The unfolding S_x (Fig. 5) is shown lightly shaded; S_y is shown dotted. Their common kernel $K_x = K_y$ is the central dark region.*

such an embedding of S_x, and use K_R to denote K_x for all points $x \in R$. Similarly, define K_ε for an edgelet ε.

3.3. How Many Different Unfoldings Are There?. For the algorithms described in this paper, it will be important to bound the number of different possible combinatorial structures of star unfoldings, as we vary the position of source point x, and to compute these unfoldings efficiently (more precisely, compute their combinatorial structure plus some metric description, parameterized by the exact position of the source), as the source moves on the surface of the polytope. Two variants of this problem will be needed. In the first, we assume that the source is placed on an edge of $\mathcal{P}$, and in the second the source is placed anywhere on $\mathcal{P}$. In view of Lemma 3.5, it suffices to bound the number of edgelets and ridge-free regions, respectively.

LEMMA 3.7. *In the worst case, there are $\Theta(n^3)$ edgelets and they can be computed in $O(n^3 \log n)$ time.*

Proof. Each edge can meet a ridge of the ridge tree of a corner at most once, since ridges are shortest paths (recall that we assume that no ridge overlaps an edge—removal of this assumption does not invalidate our argument, but only adds a number of technical complications). This gives an upper bound of $n \times O(n) \times O(n)$ on the number of edge-ridge intersections and therefore on the number of edgelets. An example of a convex polytope with $\Omega(n^3)$ edgelets is relatively easy to construct by modifying the lower bound construction of Mount [Mou90].

To compute the edgelets, we construct ridge trees from every corner in $n \times O(n^2)$ time by n applications of the algorithm of Chen and Han [CH90]. The edgelets are now computed by sorting the intersections with ridges along each edge. □

LEMMA 3.8. *In the worst case, there are $\Theta(n^4)$ ridge-free regions on $\mathcal{P}$. They can be computed in $\Theta(n^4)$ time.*

Proof. The overlay of n ridge trees, one from each corner of $\mathcal{P}$, produces a subdivision of $\mathcal{P}$ in which every region is bounded by at least three edges (cf. Fig. 2). Thus, by Euler's formula, the number of regions in this subdivision is proportional to the number of its vertices, which we proceed to estimate.

By Lemma 2.4 ridges are shortest paths and therefore two of them intersect in at most two points (cf. Corollary 2.2) or overlap. In the latter case no new vertex of the subdivision is created, so we restrict our attention to the former. In particular, as there are $n \times O(n) = O(n^2)$ ridges, the total number of their intersection points is $O(n^4)$. Refining this partition further by adding the edges of $\mathcal{P}$ does not affect the asymptotic complexity of the partition, as ridges intersect edges in $O(n^3)$ points altogether (cf. Lemma 3.3). This establishes the upper bound.

It is easily checked that there are $\Omega(n^4)$ ridge-free regions in Mount's example of a polytope with $\Omega(n^4)$ shortest-path edge sequences [Mou90]. Hence there are $\Theta(n^4)$ ridge-free regions on $\mathcal{P}$ in the worst case.

The ridge-free regions can be computed by calculating the ridge tree for every corner, and overlaying the trees in each face of $\mathcal{P}$. The first step takes $O(n^3)$ time, while the second step can be accomplished in time $O((r + n^3) \log n) = O(n^4 \log n)$, where r is the number of ridge-free regions in $\mathcal{P}$, using the line-sweep algorithm of Bentley and Ottmann [BO79]. If computing the ridge-free regions is a bottleneck, the last step can be improved to $O(n^4)$ by using a significantly more complicated algorithm of Chazelle and Edelsbrunner [CE92]. (See also [CS89, Ba95].) □

3.4. Encoding Ridge Trees. In Section 3.1, we proved that the combinatorial structure of S_x is the same for all points x in a ridge-free region. As x moves in a ridge-free region, the ridge tree T_x changes continuously (in the Hausdorff metric), as a subset of $\mathcal{P}$. In this subsection, we prove an upper bound on the number of different combinatorial structures of T_x as the source point x varies over a ridge-free region or an edgelet. In fact, we are interested not so much in counting the number of distinct ridge trees, as in representing all possible ridge trees compactly, in order, for example, to extract all vertices that ever occur in the ridge trees. Apart from being interesting in their own right, these results are needed in the algorithms described in Sections 5–7.

Let R be a ridge-free region, and let x be a point in R. By Theorem 3.4, T_x is the Voronoi diagram $\mathcal{V}_x$ of the set $X = U(x)$ of images of x clipped to lie within K_R. Since ridge vertices do not lie on ∂S_x, all changes in T_x, as x varies in R, can be attributed to changes in $\mathcal{V}_x$. Thus it suffices to distinguish distinct combinatorial structures of Voronoi diagrams $\mathcal{V}_x$, $x \in R$. Here by "combinatorial complexity" we mean an enumeration of vertices, edges, and regions of the Voronoi diagram, together with incidence relations between them.

Let $X = U(x) = \{x_1, \ldots x_n\}$. For each $x_i = (x_{i1}, x_{i2})$, let

$$f_i(y) = d(x_i, y) = \sqrt{(x_{i1} - y_1)^2 + (x_{i2} - y_2)^2}\,,$$

where (y_1, y_2) are the coordinates of a generic point y in the plane. Let

$$f(y) = \min_{1 \le i \le n} f_i(y)$$

be the *lower envelope* of the f_i's. Then $\mathcal{V}_x$ is the same as (the 1-skeleton of) the orthogonal projection of the graph of $f(y)$ onto the y_1y_2-plane, labeled with the name(s) of the function(s) attaining the minimum at each point.

We introduce an orthogonal coordinate system in R and let x have coordinates (s, t) in this system. Then positions of x_i are linear functions of s, t of the form

$$(1) \qquad \begin{pmatrix} x_{i1} \\ x_{i2} \end{pmatrix} = \begin{pmatrix} a_i \\ b_i \end{pmatrix} + \begin{pmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{pmatrix} \begin{pmatrix} s \\ t \end{pmatrix},$$

where (a_i, b_i) are coordinates of x_i when x is at the origin of the (s, t) coordinate system in R, and θ_i defines the orientation of the ith image of R in the plane.

We now regard f and f_i's as 4-variate functions of s, t, y_1, y_2, and denote by M_R the (labeled) projection of the graph of f onto the (s, t, y_1, y_2)-plane. All possible combinatorial structures of $\mathcal{V}_x$, as x varies over the entire plane, are obviously encoded in M_R, as the diagram for $x = (\alpha, \beta)$ is simply the (one-skeleton) of the cross section of M_R by the 2-flat $s = \alpha$, $t = \beta$. Let

$$(2) \qquad g_i(s, t, y_1, y_2) = (f_i(y_1, y_2))^2 - (s^2 + t^2 + y_1^2 + y_2^2)\,.$$

Using (1) we obtain

$$(3) \qquad \begin{aligned} g_i(s, t, y_1, y_2) &= C_i^{(0)} + C_i^{(1)} y_1 + C_i^{(2)} y_2 + C_i^{(3)} s + C_i^{(4)} t + C_i^{(5)} s y_1 + C_i^{(6)} s y_2 \\ &\quad + C_i^{(7)} t y_1 + C_i^{(8)} t y_2, \end{aligned}$$

where, for each i, the C_i's are constants that depend solely on a_i, b_i, and θ_i. Let $g(s, t, y_1, y_2)$ denote the lower envelope of the g_i's. Since $g(s, t, y_1, y_2) = (f(s, t, y_1, y_2))^2 - (s^2 + t^2 + y_1^2 + y_2^2)$, the projection of the graph of g is the same as M_R. Let

$$(4) \qquad v_1 = sy_1,\ v_2 = sy_2,\ v_3 = ty_1,\ v_4 = ty_2$$

and set

$$\begin{aligned}
\bar{g}_i(s,t,y_1,y_2,v_1,v_2,v_3,v_4) &= C_i^{(0)} + C_i^{(1)} y_1 + C_i^{(2)} y_2 + C_i^{(3)} s + C_i^{(4)} t \\
&\quad + C_i^{(5)} v_1 + C_i^{(6)} v_2 + C_i^{(7)} v_3 + C_i^{(8)} v_4. \qquad (5)
\end{aligned}$$

Then every face of the graph of g is the intersection of the lower envelope of $\bar{g}_i$'s with the surface defined by the equations (4). Since each $\bar{g}_i$ is a 8-variate linear function, by the Upper Bound Theorem for Convex Polyhedra, the graph of their lower envelope has $O(n^4)$ faces of all dimensions (and in fact can be triangulated by using $O(n^4)$ simplices). Hence, the number of faces in M_R is also $O(n^4)$. Using the algorithm of Brönnimann et al. [BCM94], all the faces of this lower envelope, and thus all the edges and vertices of M_R, can be computed in $O(n^4)$ time. (Using the triangulation mentioned above, one could compute a representation of all faces of M_R at the same time, by intersecting each simplex of the triangulation with the surface (4). Our algorithms will only need the edges and vertices of M_R, however.) Putting everything together, we conclude

LEMMA 3.9. *All the ridge trees for source points lying in a ridge-free region R can be encoded in a single lower envelope M_R whose combinatorial complexity is $O(n^4)$. Moreover, the edges and vertices of M_R can be computed in time $O(n^4)$.*

Remark. The only reason for assuming in the above analysis that x stays away from the boundary of R was to ensure that the vertices of the Voronoi diagram avoid the boundary of K_R. However, it is easy to verify that when x is allowed to vary over the closure of R, Voronoi vertices never cross the boundary of K_R, but may touch it in limiting configurations. Thus the same analysis applies in that case as well.

If the source point moves along an edgelet ε rather in a ridge-free region, we can obtain a bound on the number of different combinatorial structures of ridge trees by setting $t = 0$ in (1). Proceeding in the same way as above, each g_i now becomes

$$g_i(s,y_1,y_2) = C_i^{(0)} + C_i^{(1)} y_1 + C_i^{(2)} y_2 + C_i^{(3)} s + C_i^{(5)} s y_1 + C_i^{(6)} s y_2 \,.$$

Again, we define g as the lower envelope of g_i's, and the subdivision M_ε as the labeled projection of the graph of the lower envelope g. Let $v_1 = sy_1, v_2 = sy_2$, and set

$$\bar{g}_i(s,y_1,y_2,v_1,v_2) = C_i^{(0)} + C_i^{(1)} y_1 + C_i^{(2)} y_2 + C_i^{(3)} s + C_i^{(5)} v_1 + C_i^{(6)} v_2 \,. \qquad (6)$$

Since $\bar{g}_i$ is now a 5-variate linear function, by the Upper Bound Theorem, the number of faces in M_ε is $O(n^3)$. A similar bound was proved earlier in [GMR91]. Since the lower envelope of $\bar{g}_i$'s can be computed in $O(n^3)$ time [BCM94], the vertices and edges of M_ε can also be computed in time $O(n^3)$. Hence, we obtain

LEMMA 3.10. *All the ridge trees that occur as the source point moves in an edgelet ε can be represented as an envelope M_ε of n trivariate functions; its complexity is $O(n^3)$, and its edges and vertices can be computed in $O(n^3)$ time.*

Remark. Only a portion of M_R (resp. M_ε) is relevant to our analysis of ridge trees. Recall that it encodes the diagrams $\mathcal{V}_x$ for all x. However, we are only interested in $x \in R$ (resp. $x \in \varepsilon$), and not all of $\mathcal{V}_x$, but just the portion contained in K_R (resp. K_ε). Hence only those points $(s,t,y_1,y_2) \in M_R$ (resp. $(s,y_1,y_2) \in M_\varepsilon$) are relevant for which $(s,t) \in R$ (resp. $s \in \varepsilon$) and $(y_1,y_2) \in K_R$ (resp. $(y_1,y_2) \in K_\varepsilon$). When we make use of the information stored in M_R and M_ε at a later point in the computation, we will have to "filter out" irrelevant features. This issue is addressed later (Section 5).

4. Edge Sequences Superset. In this section we describe an $O(n^6)$ algorithm for constructing a superset of the shortest-path edge sequences, which is both more efficient and conceptually simpler than previously suggested procedures, and which produces a smaller set of sequences.

Observe that all shortest-path edge sequences are realized by pairs of points lying on edges of $\mathcal{P}$—any other shortest path can be contracted without affecting its edge sequence so that its endpoints lie on edges of $\mathcal{P}$. Let x be a generic point lying on an edgelet ε. As mentioned in Section 3.1, the pasting tree Π_x contains all shortest path edge sequences that emanate from x. Moreover by Lemma 3.5, Π_x is independent of choice of x in ε, therefore we will use Π_ε to denote Π_x for any point $x \in \varepsilon$. The set of $O(n^3)$ pasting trees $\{\Pi_\varepsilon \mid \varepsilon \text{ is an edgelet}\}$, each of size $O(n^2)$, contains an implicit representation of a set of $O(n^6)$ sequences ($O(n^5)$ of which are maximal in this set), which includes all shortest-path edge sequences that emanate from generic points.

Algorithm 1
Sequence Trees

```
for each edge e of P do
    Σ_e = ∅.
    for each edgelet endpoint v ∈ e do
        Compute shortest-path edge sequences Σ_v emanating from v.
        Σ_e = Σ_e ∪ Σ_v.

    for each edgelet ε ⊂ e do
        Compute Π_ε.
        for each maximal sequence σ ∈ Π_ε do
            Σ_e = Σ_e ∪ {σ}.

    T_e = the trivial sequence tree consisting of just e.
    for each sequence σ ∈ Σ_e do
        Traverse σ, augmenting T_e.
        Stop if σ visits the same edge twice.

    T_e is the sequence tree containing shortest path edge sequences
        emanating from e.
```

Hence, we can compute a superset of shortest path edge sequences in three steps: First, partition the point on the edges of $\mathcal{P}$ into $O(n^3)$ edgelets in time $O(n^3)$ as described in Lemma 3.7. Second, compute shortest path edge sequences from the endpoints of each edgelet, using Chen and Han's shortest path algorithm. Next, compute the star unfolding from a point in each edgelet, again using the shortest path algorithm. The total time spent in the last two steps is $O(n^5)$. Finally, this representation of edge sequences is transformed into $O(n)$ sequence trees, one for each edge (cf. Section 2.2); see Algorithm 1 for pseudocode. For each pasting tree Π_ε, we traverse Π_ε from each of its leaves separately, so we spend $O(n^3)$ time per pasting tree. Hence, the total time spent is $O(n^6)$. We thus obtain

Theorem 4.1. *Given a convex polytope in $\mathbb{R}^3$ with n vertices, one can construct, in time $O(n^6)$, $O(n)$ sequence trees that store a set of $O(n^6)$ edge sequences, which include all shortest path edge sequences of $\mathcal{P}$.*

Remarks. (i) Note that our algorithm uses nothing more complex than the algorithm of Chen and Han for computing shortest paths from a fixed point, plus some sorting and tree traversals. It achieves an improvement over previous algorithms mainly by reorganizing the computation around the star unfolding.

(ii) The sequence-tree representation for just the shortest-path edge sequences is smaller by a factor of n^2 than our estimate on the size of the set produced by Algorithm 1 (cf. Section 2.2), but computing it efficiently seems difficult. In addition, it is not clear how far the actual output of our algorithm is from the set of all shortest-path edge sequences. We have a sketch of a construction for a class of polytopes that force our algorithm to produce $\Omega(n^5)$ *non*-shortest-path edge sequences.

5. Exact Set of Shortest-Path Edge Sequences. In this section, we present an $O(n^3\beta(n)\log n)$ algorithm for computing the exact set of maximal shortest-path edge sequences emanating from an edgelet. Here $\beta(\cdot)$ is an extremely slowly-growing function, asymptotically smaller than $\log^* n$. Running this algorithm for all edgelets of $\mathcal{P}$, the exact set of maximal shortest-path edge sequences can be computed in time $O(n^6\beta(n)\log n)$, which is a significant improvement over Schevon and O'Rourke's $O(n^9\log n)$ algorithm [SO89].

Let ε be an edgelet. For the purposes of this section we consider S_x embedded in the plane so that $K_x = K_\varepsilon$ does not move as x varies along ε. So on the one hand, K_ε is a fixed simple n-gon in the plane and on the other hand it is a complex put together out of $O(n^2)$ convex pieces of faces of $\mathcal{P}$. By analogy with S_x, we call these pieces *plates of* K_ε. A plate of K_ε is fully contained in a plate of S_x, for any $x \in \varepsilon$. This latter plate may change its shape as x moves in ε, but always corresponds to the same node of the pasting tree $\Pi_x = \Pi_\varepsilon$. Moreover, there is at most one plate of K_x in a plate of S_x, as a plate of K_x is obtained from a convex set (a plate of S_x) contained in a simple polygon (S_x) by cutting off n triangles ($\triangle x_i p_{i-1} p_i$, for $i = 1, \ldots, n$), each by a chord ($p_{i-1}p_i$).

We are interested in computing the set Σ_ε of those shortest-path edge sequences (corresponding to paths emanating from points on ε) which are maximal over all points in ε. In other words, given a sequence $\sigma \in \Sigma_\varepsilon$, there is a point $x \in \varepsilon$ and a shortest path starting from x that traverses σ; and there is no point on ε from which there is a shortest path that traverses an edge sequence that is an extension of σ. Recall that each sequence in Σ_ε corresponds to a path in the pasting tree Π_ε, originating from one of its leaves. Each leaf of Π_ε corresponds to a triangular plate incident to one of the n images of x. Let $\Sigma_{\varepsilon,i} \subseteq \Sigma_\varepsilon$ denote the set of edge sequences that originate from the ith leaf of Π_ε. If a sequence $\sigma \in \Sigma_{\varepsilon,i}$ is realized by a shortest path π emanating from $x \in \varepsilon$, then π leaves x between xp_{i-1} and xp_i, and it corresponds to a segment in S_x emanating from x_i, the ith image of x. The area swept by all these segments (for a fixed x) is exactly the ith peel $P_{x,i}$. $P_{x,i}$ consists of the triangle $\triangle x_i p_{i-1} p_i$ and the "remainder" $\hat{P}_{x,i}$, which is the portion of the ith peel that lies in $K_x = K_\varepsilon$. If we concentrate on *maximal* shortest paths contained in $P_{x,i}$, it is sufficient to consider those paths that end in $\hat{P}_{x,i}$, as any path that ends in $\triangle x_i p_{i-1} p_i$ can be extended to a point in $\hat{P}_{x,i}$ while remaining a shortest path. Put $C_i = \bigcup_{x\in\varepsilon} \hat{P}_{x,i}$; see Fig. 9a.

LEMMA 5.1. *Let $p, p' \in C_i$ be two points lying in the same plate of K_ε. Suppose $p \in \hat{P}_{x,i}$ and $p' \in \hat{P}_{x',i}$ for some $x, x' \in \varepsilon$. Then the edge sequences corresponding to the segments $x_i p$ and $x'_i p'$ are the same. The edge sequences are different if $p, p' \in C_i$ are contained in different plates of K_ε.*

Proof. Let v be the node of Π_ε corresponding to the plate of K_ε that contains p and p'. Since there is a unique path from the ith leaf to v in Π_ε, the lemma follows. Different plates of K_ε, as observed above correspond to different nodes of Π_ε, and thus to different sequences of edges, essentially by definition of Π_ε. □

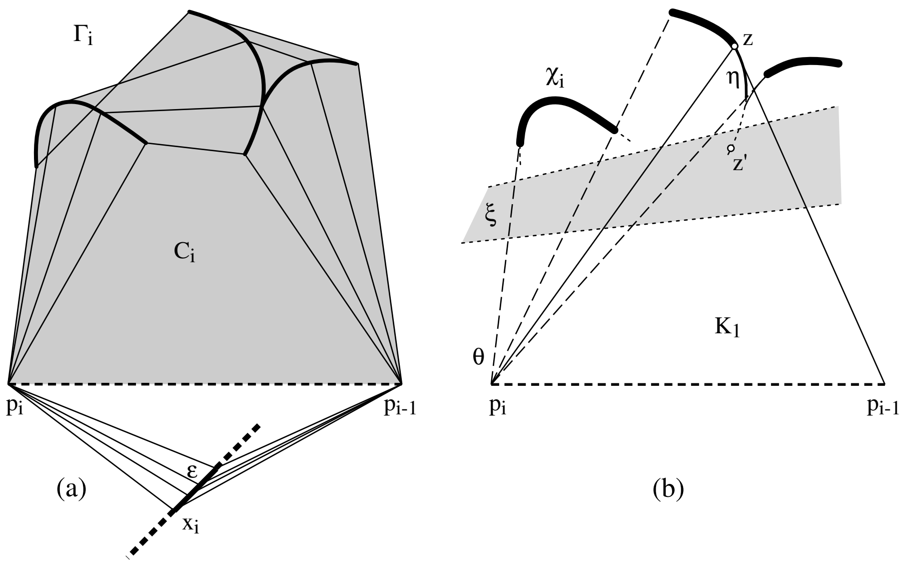

FIG. 9. *(a) C_i is the union of the peel remainders $\hat{P}_{x,i}$. Γ_i comprise the arcs traced by the ridge vertices (shown dark). (b) γ_i is the θ-monotone upper envelope (solid) of Γ_i with respect to p_i; several key radial lines from p_i are shown (dashed). γ_i is a superset of the "outer" envelope χ_i (dark); here the arc $\eta \in \gamma_i \setminus \chi_i$. A covering triangle $\triangle z p_{i-1} p_i$ whose apex z covers $z' \in (\cup\Gamma_i) \cap \xi$ is shown (solid).*

By the above lemma, each point of C_i *determines* a unique shortest-path edge sequence of $\Sigma_{\varepsilon,i}$, and all points of C_i lying in the same plate of K_ε determine the same shortest-path edge sequence of $\Sigma_{\varepsilon,i}$. We mark a node of Π_ε if the corresponding plate of K_ε intersects C_i. Let $\Pi_{\varepsilon,i}$ be the minimal subtree of Π_ε rooted at the ith leaf and containing all marked nodes. Then there is a one-to-one correspondence between the sequences of $\Sigma_{\varepsilon,i}$ and the paths in $\Pi_{\varepsilon,i}$ from its root to leaves.

LEMMA 5.2. *Let σ be an edge sequence in $\Sigma_{\varepsilon,i}$ realized by a shortest path originating from point $x \in \varepsilon$. Then there is vertex p of $\hat{P}_{x,i}$ such that the segment $x_i p$ realizes the sequence σ.*

Proof. By Lemma 5.1, σ is realized by all points in the intersection of a unique plate ξ of K_ε with C_i. Choose a point $p \in \hat{P}_{x,i} \cap \xi$ such that $|x_i p|$ is maximum among all such points, where $|\ldots|$ denotes the Euclidean length of a segment. (Notice that the maximum must be achieved, for otherwise there is a shortest path from x_i whose sequence extends that of σ.) If p is not a vertex of $\hat{P}_{x,i}$, then we can choose a point $p' \in \hat{P}_{x,i} \cap \xi$ in a sufficiently small neighborhood of p such that $|x_i p'| > |x_i p|$, a contradiction. Hence, we can assume that p is a vertex of $\hat{P}_{x,i}$, as desired. ☐

In view of this lemma, we can restrict our attention to the points of C_i that correspond to the vertices of $\hat{P}_{x,i}$, for any $x \in \varepsilon$. Recall each vertex of $\hat{P}_{x,i}$ is the image of a ridge vertex. A typical ridge vertex v is incident to three open peels c_j, c_k, c_ℓ; if v has degree more than three and exists at more than just a discrete set of positions of $x \in \varepsilon$, replace the triple of incident peels by a larger tuple in the following discussion. As x moves along ε, the vertex traces an algebraic curve $v = v(x)$ in K_ε. Let a *lifetime* of a ridge vertex v be a maximal connected interval $\varepsilon' \subseteq \varepsilon$ for which $x \in \varepsilon'$ implies that v is a vertex of T_x. Let Γ_i be the set of arcs traced out by ridge vertices that appear on the boundary of $\hat{P}_{x,i}$ during their lifetimes (Fig. 9a);

set $n_i = |\Gamma_i|$. It can be verified that the arcs in Γ_i corresponding to a ridge vertex, defined by the triple c_j, c_k, c_ℓ, are the projections onto the xy-plane of those edges of the subdivision M_ε, defined in Section 3.4, along which g_j, g_k, g_ℓ simultaneously appear on the lower envelope g. (As we mentioned in Section 3.4, M_ε may contain "irrelevant" features. In particular, we must first truncate each aforementioned arc so that it corresponds to positions of the source on ε. Secondly, we must verify that the Voronoi diagram vertex corresponding to the arc indeed yields a ridge vertex. It is sufficient to check, for a single point of the curve traced out by the vertex as x ranges over ε, that it lies inside K_ε, as a ridge vertex cannot leave K_ε. This is easily accomplished by one point-location query per arc.)

We have previously observed that the maximal sequences of $\Pi_{\varepsilon,i}$ are necessarily realized by points on $\partial C_i \setminus p_{i-1}p_i$. In particular, points that lie in the interior of C_i can be safely disregarded. We will now apply a similar procedure to points of $\cup\Gamma_i$. If we introduce polar coordinates with p_i as the origin, each arc $\eta_j \in \Gamma_i$ can be regarded as a univariate (partial) function $r = \eta_j(\theta)$ (split η_j into a constant number of θ-monotone arcs if it is not θ-monotone; this is possible since η_i is a portion of an algebraic curve of small degree). Consider the graph of the upper envelope γ_i of the functions $\eta_j(\theta)$ (Fig. 9b). Since each arc in Γ_i is algebraic of constant degree, the upper envelope γ_i has $O(n_i\beta(n_i))$ breakpoints [ASS89]; here $\beta(k) = 2^{\alpha^s(k)}$, s is a constant depending on the maximum degree of arcs in Γ_i, and $\alpha(\cdot)$ is the inverse Ackermann function. Using a divide-and-conquer approach the upper envelope can be computed in time $O(n_i\beta(n_i)\log n_i)$; see [SA95].

We will now show that tracing γ_i through K_ε is sufficient for computing $\Pi_{\varepsilon,i}$—there is no need to examine the entire $\cup\Gamma_i$.

LEMMA 5.3. *Each sequence in $\Sigma_{\varepsilon,i}$ is determined by a point on γ_i.*

Proof. We will show in fact that points on a subset χ_i of γ_i suffice to determine all sequences. Say a point z "covers" a point $z' \neq z$ if the triangle $\triangle z p_{i-1} p_i$ contains z'. Call the set of points of $\cup\Gamma_i$ not covered by any point of $\cup\Gamma_i$ its *outer envelope* χ_i. It is evident that $\chi_i \subseteq \gamma_i$; see Fig. 9b.

Suppose there is a sequence $\sigma \in \Sigma_{\varepsilon,i}$ not realized by any point on χ_i. Let ξ be the plate of K_ε corresponding to σ. Thus some arc of Γ_i meets ξ, but χ_i does not. Hence every point of $(\cup\Gamma_i) \cap \xi$ is covered by a point of χ_i. Pick a point $z \in \chi_i$ that covers some point $z' \in (\cup\Gamma_i) \cap \xi$; see Fig. 9b. By the choice of z', there is an $x \in \varepsilon$ so that the segment xz' corresponds to a shortest path on $\mathcal{P}$. Consider $K_\varepsilon \setminus \xi$. It consists of two or three simple polygons, one of which, say K_1, is incident to $p_{i-1}p_i$ (the case when ξ touches $p_{i-1}p_i$ is slightly different and can be handled by an easier argument). We claim that z does not lie in K_1. Indeed, $\triangle z' p_{i-1} p_i$ is such that both $p_{i-1}z'$ and $p_i z'$ enter ξ and remain there. As $\triangle z p_{i-1} p_i$ contains $\triangle z' p_{i-1} p_i$, removal of ξ separates K_ε, and $z \notin \xi$, both $p_{i-1}z$ and $p_i z$ must cross ξ and exit it. As $z \in \chi_i$, there is an $x' \in \varepsilon$ so that the segment $x'z$ is (the image of) a shortest path. However, since $z \notin K_1$, this path crosses ξ, so the edge sequence it traverses is an extension of σ, contradicting maximality of σ.

We have shown that each sequence is realized by a point of χ_i, and therefore of γ_i. □

If a plate of K_ε is intersected by an edge of γ_i, we call the corresponding node of Π_ε *visited* by γ_i. The above lemma implies that the minimal subtree containing the node corresponding to x_i and all nodes visited by γ_i is the same as $\Pi_{\varepsilon,i}$. It is thus sufficient to determine the nodes of Π_ε visited by γ_i.

Note that γ_i is θ-monotone with respect to p_i, and therefore γ_i lies in the portion

Algorithm 2
Exact Edge Sequences

```
Form edgelets.
for each edgelet ε do
    Compute M_ε.
    Γ := Set of projections of edges in M_ε.
    Eliminate irrelevant features from Γ.
    for each image x_i do
        Γ_i := Arcs of Γ at which g_i appears on the lower envelope.
        Compute upper envelope γ_i of Γ_i.
        Compute and triangulate K_ε[p_i].
        Compute the connected arcs ξ_1, ... , ξ_u.
        for each △ in the triangulation do
            Compute Π_△ := Π_ε ∩ △.
            Compute Π^1_△, Π^2_△.
            for each arc ξ_j ⊂ △ do
                Find nodes of Π^1_△, Π^2_△ visited by ξ_j.
```

$K_\varepsilon[p_i]$ of K_ε visible from p_i. Compute $K_\varepsilon[p_i]$, in linear time [EA81], and triangulate $K_\varepsilon[p_i]$ into a linear number of triangles all incident to p_i. Now partition γ_i into connected portions $\xi_1, \ldots, \xi_u$, each fully contained in one of these triangles. This can be done in time proportional to the number of breakpoints in γ_i and the number of triangles involved and produces at most $O(n)$ extra arcs, as γ_i is θ-monotone (with respect to p_i) and thus crosses each segment separating consecutive triangles in at most one point. Since each triangle $\triangle$ is fully contained in K_ε and thus encloses no images of a vertex of $\mathcal{P}$, the set of plates of Π_ε met by $\triangle$ corresponds to a subtree $\Pi_\triangle$ of Π_ε of linear size, with at most one vertex of degree 3 and all remaining vertices of degree at most 2. Hence, $\Pi_\triangle$ can be covered by two simple paths $\Pi^1_\triangle, \Pi^2_\triangle \subseteq \Pi_\triangle$, and they can be computed in linear time. For each $\xi_j \subset \triangle$, we determine the furthest node that ξ_j reaches in $\Pi^1_\triangle, \Pi^2_\triangle$ by binary search. Recall that ξ_j consists of a number of algebraic arcs of constant degree. An intersection between ξ_j and a plate of K_ε can be detected in $O(1)$ time per such arc, so the binary search requires only $O(\log n)$ time per arc. The total time spent is thus $O(n_i\beta(n)\log n + n^2)$ over all triangles of $K_\varepsilon[p_i]$. Repeating this procedure over all n leaves of Π_ε, the total time spent in computing Σ_ε is $O(n^3\beta(n)\log n)$, as $\sum_i n_i = O(n^3)$ by Lemma 3.10. The above processing is repeated for each of the $O(n^3)$ edgelets ε. This completes the description of the algorithm. It is summarized in Algorithm 2.

Theorem 5.4. *The exact set of all shortest-path edge sequences on the surface of a 3-polytope on n vertices can be computed in $O(n^6\beta(n)\log n)$ time, where $\beta(n) = o(\log^* n)$ is an extremely slowly-growing function.*

6. Geodesic Diameter. In this section we present an $O(n^8 \log n)$ time algorithm for computing the geodesic diameter of $\mathcal{P}$. As mentioned in the introduction, this question was first investigated by O'Rourke and Schevon [OS89] who presented an $O(n^{14}\log n)$ time algorithm for computing it. Their algorithm relies on the following proposition:

Lemma 6.1 (O'Rourke and Schevon [OS89]). *If a pair of points $x, y \in \mathcal{P}$ realizes the diameter of $\mathcal{P}$, then either x or y is a corner of $\mathcal{P}$, or there are at least* *five distinct shortest paths between x and y.*

Lemma 6.1 suggests the following strategy for locating all diametral pairs. We first dispose of the possibility that either x or y is a corner in $n \times O(n^2) = O(n^3)$ time just as in [OS89]. Next, we fix a ridge-free region R and let M_R be the subdivision defined in Section 3.4. We need to compute all pairs of points $x \in \mathrm{cl}\,(R)$ and $y^* \in K_x$ such that there are at least five distinct shortest paths between x and y, with $U(y) = y^*$. By a result of Schevon [Sch89], such a pair x, y can be a diametral pair only if it is the only pair, in a sufficiently small neighborhood of x and y, with at least five distinct shortest paths between them. Such a pair of points corresponds to a vertex of M_R. Hence, we use the following approach.

We first compute, in $O(n^4)$ time, all ridge-free regions of $\mathcal{P}$ (cf. Lemma 3.8). Next, for each ridge-free region R, we compute K_R, vertices of M_R, and $f(v)$ for all vertices of M_R (recall that $f(v)$ is the shortest distance from v to any source image; cf. Section 3.3). Next, for each vertex $v = (s, t, y_1, y_2)$ of M_R, we determine whether (s, t) lies in the closure of R and $(y_1, y_2) \in K_R$. If the answer to both of these questions is 'yes', we add v to the list of candidates for diametral pairs. (This step is exactly the elimination of "irrelevant features" mentioned at the end of Section 3.4. Once the two conditions are verified, we know that (s, t) and (y_1, y_2) correspond to actual points x, y on $\mathcal{P}$ and $f(v)$ is exactly $d(x, y)$.)

Finally, among all diametral candidate pairs, we choose a pair that has the largest geodesic distance. See Algorithm 3 for the pseudocode.

For each ridge-free region R, K_R can be computed in time $O(n^2)$ and preprocessed for planar point location in additional $O(n \log n)$ time using the algorithm of Sarnak and Tarjan [ST86]. (Once again, recall that we treat K_R as a simple polygon and use (y_1, y_2)-coordinate system there.) By Lemma 3.9, vertices of M_R and $f(v)$, for all vertices of M_R, can be computed in time $O(n^4)$. We spend $O(\log n)$ time for point location at each vertex of M_R, so the total time spent is $O(n^8 \log n)$.

Algorithm 3
Geodesic Diameter

```
for each corner c of P do
    Construct the ridge tree T_c with respect to c.
    for each vertex v of T_c do
        Add d(c, v) to the list of diameter candidates.
Compute the ridge-free regions.
for each ridge-free region R do
    Compute M_R and f(v) for all vertices v ∈ M_R.
    Compute S_x for some x ∈ R.
    Construct K_R = K_x.
    Preprocess K_R for point location queries.
    Preprocess cl (R) for point location queries.
    for each vertex v = (s, t, y_1, y_2) of M_R do
        if (s, t) ∈ cl (R) and (y_1, y_2) ∈ K_x then
            Add f(v) = d((s, t), (y_1, y_2)) to the list of diameter candidates.
Find a diametral candidate pair with the maximum geodesic distance.
```

Theorem 6.2. *The geodesic diameter of a convex polytope in $\mathbb{R}^3$ with n vertices can be computed in time $O(n^8 \log n)$.*

7. Shortest-Path Queries. In this section we discuss the preprocessing needed to support queries of the form: "Given $x, y \in \mathcal{P}$, determine $d(x, y)$." We assume that each face ϕ of $\mathcal{P}$ has its own coordinate system (e.g., a vertex of ϕ is regarded as the origin and the two edges of ϕ incident to it are regarded as the two axes), and that a point $p \in \mathcal{P}$ is specified by the face ϕ containing p and by the ϕ-coordinates of p. Two variants of the query problem are considered: (1) no assumption is made about x and y and (2) x is assumed to lie on an edge of $\mathcal{P}$.

Our data structure is based on the following observations. Let $x, y \in \mathcal{P}$ be two query points. Suppose $x = (s, t)$ is a generic point lying in a ridge-free region R and $y^* = (y_1, y_2)$ is an image of y in S_x. If y^* lies in the kernel K_R, then

$$d(x, y) = f(s, t, y_1, y_2) = (g(s, t, y_1, y_2) + (s^2 + t^2 + y_1^2 + y_2^2))^{1/2},$$

where

$$g(s, t, y_1, y_2) = \min_{1 \le i \le n} \bar{g}_i(s, t, y_1, y_2, v_1, v_2, v_3, v_4),$$

as defined in (2), (4), and (5). Let H_R be the set of hyperplanes in $\mathbb{R}^9$ corresponding to the graphs of $\bar{g}_i$'s (cf. (5))

$$h_i : v_5 = C_i^{(0)} + C_i^{(1)} y_1 + C_i^{(2)} y_2 + C_i^{(3)} s + C_i^{(4)} t + C_i^{(5)} v_1 + C_i^{(6)} v_2 + C_i^{(7)} v_3 + C_i^{(8)} v_4. \tag{7}$$

Then computing the value of $g(s, t, y_1, y_2)$ is the same as determining the first hyperplane of H_R intersected by the vertical ray emanating from the point

$$(s, t, y_1, y_2, sy_1, sy_2, ty_1, ty_2, -\infty)$$

in the positive v_5-direction. The desired value is $(v_5 + s^2 + t^2 + y_1^2 + y_2^2)^{1/2}$, where v_5 is the v_5-coordinate of the intersection point.

On the other hand, if $y^* \notin K_x$, then it lies in one of the triangles $\triangle p_{i-1} x_i p_i$ and $d(x, y) = |x_i y^*|$. For a ridge-free region R, let κ_R denote the preimage of ∂K_R on $\mathcal{P}$, i.e., $U(\kappa_R) = \partial K_R$; The following lemma is crucial in answering queries when $y \notin U^{-1}(K_R)$.

LEMMA 7.1. *Let R be a ridge-free region or an edgelet, let ϕ be a face of $\mathcal{P}$, and let Δ be a connected component of $\phi \setminus \kappa_R$ whose image is not contained in K_R. Then the sequence of edges traversed by the shortest path $\pi(x, y)$ is independent of the choice of $x \in R$ and $y \in \Delta$.*

Proof. For the sake of a contradiction, suppose there are two points $y, y' \in \Delta$ such that the sequences of edges traversed by $\pi(x, y')$ and $\pi(x, y'')$ are distinct. Then there must exist a point $y \in y'y''$ with two shortest paths to x—to obtain such a point, move y from one end of $y'y''$ to the other and observe that the shortest path from x to y changes continuously and maintains the set of edges of $\mathcal{P}$ that it meets, *except* at points y with more than one shortest path to x. Thus $y \in T_x$, so $U(y) \subset U(T_x) \subset K_R$. However, the segment $y'y'' \subset \Delta$ as Δ is convex, implying $U(y) \subset U(\Delta) \subset S_x \setminus K_R$—a contradiction.

Similarly, if $x', x'' \in R$ are such that the paths connecting these two points to $y \in \Delta$ traverse different edge sequences, there must exist $x \in x'x''$ which is connected to y by two shortest paths, again forcing y onto T_x and yielding a contradiction. The lemma follows easily. □

Data structure.. Based on the above observations, we can preprocess $\mathcal{P}$ as follows. We partition every face ϕ of $\mathcal{P}$ into ridge-free regions in time $O(n^4)$ (see Lemma 3.8), and preprocess the resulting subdivision of ϕ for planar point location queries, using any standard algorithm [ST86]. The queries would use ϕ-coordinates. The total time spent in this step is $O(n^4 \log n)$.

Let R be a fixed ridge-free region. We construct the following data structures for R. Choose an arbitrary point $x \in R$. Compute $K_x = K_R$ and the connected components of $\phi \setminus \kappa_R$, for each face ϕ of $\mathcal{P}$. Again, we preprocess the resulting subdivision of each face for planar point location queries, in ϕ-coordinates. We label each component Δ of $\phi \setminus \kappa_R$ as to whether it lies in $U^{-1}(K_R)$. If it does not, we choose a point $y \in \Delta$ and compute the edge sequence σ for the shortest path from x to y. By Lemma 7.1, σ is the same for all pairs $x \in R$ and $y \in \Delta$. We also compute the transformation, corresponding to the edge sequence σ, which maps the ϕ-based coordinates of points in Δ to ϕ'-coordinates of the face of $\mathcal{P}$ containing R. This corresponds to laying out in the plane the faces prescribed by σ from ϕ' to ϕ so that $d(x, y)$ becomes the length of the straight-line segment connecting x and y. All transformations for regions $\Delta \not\subset U^{-1}(K_R)$ can be computed in $O(n^2)$ time by a single depth-first traversal of the shortest-path sequence tree from x, computed by the algorithm of Chen and Han. If on the other hand, Δ lies in $U^{-1}(K_R)$, the exact sequence of edges traversed by a shortest path from $x \in R$ to $y \in \Delta$ depends on the choice of x and y; the structure for determining it is described below. (Note that such sets Δ correspond exactly to "plates of K_R," as in Section 5.) However, in this case, any $y \in U^{-1}(K_R)$ has a unique image $y^* \in K_R$, so for each $\Delta \subset U^{-1}(K_R)$ we compute the coordinate transformation U from the ϕ-coordinates, where ϕ is the face of $\mathcal{P}$ containing Δ, to the coordinates in the planar embedding of K_R (they were referred to as (y_1, y_2)-coordinates in Section 3.4). The sequences σ and the coordinate transformations U, for all $\Delta \subset U^{-1}(K_R)$, can be computed in $O(n^2)$ time, by performing a depth first search on Π_x (each node of Π_x corresponds to a connected component Δ).

Next, let H_R be the set of hyperplanes defined in (7). We preprocess H_R into a data structure, so that the first hyperplane of H intersected by a vertical ray emanating from a point with $v_5 = -\infty$ can be computed efficiently. Matoušek and Schwarzkopf [MS93] (see also [AM92]) have proposed such a data structure, which given a parameter $n \leq u \leq n^4$, can preprocess H_R, in time $O(u^{1+\delta})$, into a data structure of size $O(u^{1+\delta})$, so that a ray shooting query can be answered in time $O(\frac{n}{u^{1/4}} \log n)$. This completes the description of the data structures for R. We construct these data structures for each ridge-free region R.

Since there are $O(n^4)$ ridge-free regions, the total time spent in constructing the data structures is $O(n^4(n^2 + u^{1+\delta}))$.

Answering a query.. Let $x, y \in \mathcal{P}$ be a query pair. Let ϕ_x, ϕ_y be the faces of $\mathcal{P}$ containing x and y, respectively. Assume first that x is a generic point. By locating x in the point location data structure for ϕ_x, we identify in $O(\log n)$ time the ridge-free region R that contains x. Next, we determine the connected component Δ of $\phi_y \setminus \kappa_R$ that contains y, by point location in ϕ_y. If $\Delta \cap U^{-1}(K_R) = \emptyset$, we can use the transformation stored at Δ to compute $d(x, y)$ in $O(1)$ time. If $\Delta \subset U^{-1}(K_R)$, using the second data structure, we compute the first hyperplane h of H hit by the ray emanating from $(a_x, b_x, a_y, b_y, a_x a_y, a_x b_y, b_x a_y, b_x b_y, -\infty)$ in the $+v_5$-direction, where (a_x, b_x) and (a_y, b_y) are the coordinates of x and y, respectively. The coordinates of x are in the ϕ_x-coordinate system and the coordinates of y are in the coordinates system associated with the unfolding of K_R–the coordinate transformation from ϕ_y to (y_1, y_2) is stored at Δ. Once we know h, $d(x,y) = (g(a_x, b_x, a_y, b_y) + a_x^2 + a_y^2 + b_x^2 + b_y^2)^2$ can be computed in $O(1)$ time. The total time required is $O((n/u^{1/4})\log n)$.

Finally, if x is not a generic point then, as mentioned in the remark following Lemma 3.9, we can use the data structures of any of the ridge-free regions whose boundaries contain x. It is easy to see by a continuity argument that all shortest paths from such a point are encoded equally well in the data structures of all the ridge-tree regions touching x.

Hence, setting $u = n^2 m$, we can conclude

THEOREM 7.2. *Given a polytope $\mathcal{P}$ in $\mathbb{R}^3$ with n vertices and a parameter $1 \le m \le n^2$, one can construct, in time $O(n^6 m^{1+\delta})$ for any $\delta > 0$, a data structure of size $O(n^6 m^{1+\delta})$, so that $d(x,y)$ for any two points $x, y \in \mathcal{P}$ can be computed in time $O((\sqrt{n}/m^{1/4})\log n)$. Constants of proportionality depends on δ.*

If x always lies on an edge, then H is a set of hyperplanes in $\mathbb{R}^6$, so the query time of the analogous vertical ray-shooting data structure in six dimensions is $O(n/u^{1/3}\log n)$ for $n \le u \le n^2$. Moreover, we have to construct only $O(n^3)$ different data structures, one for each edgelet, so we can conclude

THEOREM 7.3. *Given a polytope $\mathcal{P}$ in $\mathbb{R}^3$ with n vertices and a parameter $1 \le m \le n$, one can construct in time, $O(n^5 m^{1+\delta})$ for any $\delta > 0$, a data structure of size $O(n^5 m^{1+\delta})$, so that for any two points $x, y \in \mathcal{P}$ such that x lies on an edge of $\mathcal{P}$ one can compute $d(x,y)$ in time $O((n/m)^{1/3}\log^2 n)$.*

8. Discussion and Open Problems. We have shown that use of the star unfolding of a polytope leads to substantial improvements in the time complexity of three problems related to shortest paths on the surface of a convex polytope: finding edge sequences, computing the geodesic diameter, and distance queries. Moreover, the algorithms are not only theoretical improvements, but also, we believe, conceptual simplifications. This demonstrates the utility of the star unfolding.

We conclude by mentioning some open problems

1. Can one obtain an upper bound on the number of different combinatorial structures of ridge trees better than $O(n^4)$? Such an improvement will yield a similar improvement in the time complexities of diameter and exact shortest path edge sequences algorithms.
2. Can one answer a shortest path query faster if *both* x and y lie on some edge of $\mathcal{P}$? This special case is important for planning paths among convex polyhedra (see Sharir [Sha87]).

Acknowledgement. The referees' comments led to significant improvements.

REFERENCES

[AAOS90] P. K. Agarwal, B. Aronov, J. O'Rourke, and C. Schevon. Star unfolding of a polytope with applications. In J. R. Gilbert and R. Karlsson, editors, *Proc. of 2nd Scandinavian Workshop on Algorithm Theory*, pp. 251–263. Springer-Verlag, July 1990. Lecture Notes in Computer Science, Vol. 447.

[Ale58] A. D. Aleksandrov. *Konvexe Polyeder.* Akademie-Verlag, Berlin, 1958. Math. Lehrbucher und Monographien.

[AM92] P. K. Agarwal and J. Matoušek. Ray shooting and parametric search. *SIAM J. Comput.*, 22:794–806, 1993.

[AO92] B. Aronov and J. O'Rourke. Nonoverlap of the star unfolding. *Discrete Comput. Geom.*, 8:219–250, 1992.

[ASS89] P. K. Agarwal, M. Sharir, and P. Shor. Sharp upper and lower bounds on the length of general Davenport-Schinzel sequences, *J. Comb. Theory, Ser. A*, 52:228–274, 1989.

[AZ67] A. D. Aleksandrov and V. A. Zalgaller. *Intrinsic Geometry of Surfaces.* American Mathematical Society, Providence, RI, 1967. Translation of the 1962 Russian original.

[Ba95] I. Balaban. An optimal algorithm for finding segments intersections, *Proc. 11th Annu. Sympos. Comput. Geom.*, 1995, pp. 211–219.

[BO79] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Trans. Comput.*, C-28:643–647, 1979.

[BCM94] H. Brönnimann, B. Chazelle, and J. Matoušek, Product range spaces, sensitive sampling, and derandomization, *Proc. 34th Annu. IEEE Sympos. Found. Comput. Sci.*, 1993, pp. 400–409.

[CR87] J. Canny and J. Reif. New lower bound techniques for robot motion planning problems. *Proc. 28th IEEE Symp. Foundations of Computer Science*, 1987, pp. 49–60.

[CE92] B. Chazelle and H. Edelsbrunner. An optimal algorithm for intersecting line segments in the plane. *J. ACM*, 39:1–54, 1992.

[CH90] J. Chen and Y. Han. Shortest paths on a polyhedron. *Proc. 6th Annu. ACM Sympos. Comput. Geom.*, pp. 360–369, 1990.

[CH91] J. Chen and Y. Han. Storing shortest paths for a polyhedron. In F. Dehne, F. Fiala, and W. W. Koczkodaj, editors, *Advances in Computing and Information - ICCI '91 International Conference Proceedings*, pp. 169–80. Springer-Verlag, 1991.

[CSY94] J. Choi, J. Sellen, and C.-K. Yap. Approximate Euclidean shortest path in 3-space, *Proc. 10th Annu. Sympos. Comput. Geom.* 1994, pp. 41–48.

[Cla87] K. Clarkson. Approximation algorithms for shortest path motion planning. *Proc. 19th ACM Symp. Theory of Computing*, 1987, pp. 56–65.

[CS89] K. Clarkson and P. Shor. Applications of random sampling in computational geometry, II, *Discrete Comput. Geom.* 4:387–421, 1989.

[EA81] H. El Gindy and D. Avis. A linear algorithm for computing the visibility polygon from a point. *Journal of Algorithms*, 2:186–197, 1981.

[Gar61] M. Gardner. *The 2nd Scientific American Book of Mathematical Puzzles and Diversions.* Simon and Schuster, New York, 1961.

[GMR91] L. Guibas and J. S. B. Mitchell and T. Roos. Voronoi diagrams of moving points in the plane, *Proc. 17th Internat. Workshop Graph-Theoret. Concepts Comput. Sci.*, Lecture Notes in Computer Science, 570, Springer-Verlag, 1991, pp. 113–125.

[HS93] J. Hershberger and S. Suri. Efficient computation of Euclidean shortest paths in the plane, *Proc. 34th Annu. IEEE Sympos. Found. Comput. Sci.*, 1993, pp. 508–517.

[HS95] J. Hershberger and S. Suri Practical Methods for Approximating Shortest Paths on a Convex Polytope in R^3. *Proc. 6th Annu. ACM-SIAM Sympos. Discrete Alg.* 1995, pp. 447–456.

[HCT89] Y.-H. Hwang, R.-C. Chang, and H.-Y. Tu. Finding all shortest path edge sequences on a convex polyhedron. In *Proc. 1st Workshop Algorithms Data Struct.*, volume 382 of *Lecture Notes in Computer Science*, pp. 251–266. Springer-Verlag, 1989.

[Kob67] S. Kobayashi. On conjugate and cut loci. In S. S. Chern, editor, *Studies in Global Geometry and Analysis*, pages 96–122. Mathematical Association of America, 1967.

[Mou85] D. Mount. On finding shortest paths on convex polyhedra. Technical Report 1495, Department of Computer Science, University of Maryland, 1985.

[Mit93] J. S. B. Mitchell. Shortest paths among obstacles in the plane. *Proc. 9th ACM Symp. Computational Geometry*, 1993, pp. 308–317.

[MMP87] J. Mitchell, D. Mount, and C. Papadimitriou. The discrete geodesic problem. *SIAM J. Comput.*, 16:647–668, 1987.

[Mou90] D. M. Mount. The number of shortest paths on the surface of a polyhedron. *SIAM J. Comput.*, 19:593–611, 1990.

[MS93] J. Matoušek and O. Schwarzkopf, J. Ray shooting in convex polytopes. *Discrete Comput. Geom.*, 10:215–232, 1993.

[OS89] J. O'Rourke and C. Schevon. Computing the geodesic diameter of a 3-polytope. *Proc. 5th ACM Symposium on Computational Geometry*, pp. 370–379, 1989.

[Pap85] C. Papadimitriou. An algorithm for shortest paths motion in three dimensions, *Info. Proc. Lett.*, 20:259–263, 1985.

[Ras90] R. Rasch. Shortest paths along a convex polyhedron. Master's thesis, University of Saarland, 1990.

[RS89] J. Reif and J. Storer. Shortest paths in Euclidean space with polyhedral obstacles, *J. ACM*, 41:1013–1019, 1994.

[Sch89] C. Schevon. *Algorithms for geodesics on polytopes.* PhD thesis, Johns Hopkins University, 1989.

[Sei81] R. Seidel. A convex hull algorithm optimal for point sets in even dimensions. Tech. Rept. 81/14, Dept. Computer Science, Univ. of British Columbia, Vancouver, 1981.

[Sha87] M. Sharir. On shortest paths amidst convex polyhedra. *SIAM J. Comput.*, 16:561–572, 1987.

[SO88] C. Schevon and J. O'Rourke. The number of maximal edge sequences on a convex polytope. In *Proc. 26th Allerton Conf. Commun. Control Comput.*, pages 49–57, October 1988.

[SO89] C. Schevon and J. O'Rourke. An algorithm for finding edge sequences on a polytope. Technical Report JHU-89/03, Dept. Comput. Sci., Johns Hopkins Univ., Baltimore, MD, February 1989.

[SS86] M. Sharir and A. Schorr. On shortest paths in polyhedral spaces. *SIAM J. Comput.*, 15:193–215, 1986.

[ST86] N. Sarnak and R. E. Tarjan. Planar point location using persistent search trees. *Comm. ACM*, 29:609-679, 1986.

[SA95] M. Sharir and P. Agarwal, *Davenport–Schinzel Sequences and Their Geometric Applications*, Cambridge University Press, New York, 1995.

[Wel85] E. Welzl. Constructing the visibility graph for n line segments in $O(n^2)$ time. *Info. Proc. Lett.*, 20:167–171, 1985.