Delaunay Triangulations in the Hilbert Metric

Auguste H. Gezalyan
Department of Computer Science, University of Maryland, College Park, MD, USA

Soo H. Kim
Wellesley College, MA, USA

Carlos Lopez
Montgomery Blair High School, Silver Spring, MD, USA

Daniel Skora
Indiana University, Bloomington, IN, USA

Zofia Stefankovic
Stony Brook University, Stony Brook, NY, USA

David M. Mount
Department of Computer Science, University of Maryland, College Park, MD, USA

Abstract

The Hilbert metric is a distance function defined for points lying within the interior of a convex body. It arises in the analysis and processing of convex bodies, machine learning, and quantum information theory. In this paper, we show how to adapt the Euclidean Delaunay triangulation to the Hilbert geometry defined by a convex polygon in the plane. We analyze the geometric properties of the Hilbert Delaunay triangulation, which has some notable differences with respect to the Euclidean case, including the fact that the triangulation does not necessarily cover the convex hull of the point set. We also introduce the notion of a Hilbert ball at infinity, which is a Hilbert metric ball centered on the boundary of the convex polygon. We present a simple randomized incremental algorithm that computes the Hilbert Delaunay triangulation for a set of n points in the Hilbert geometry defined by a convex m-gon. The algorithm runs in $O(n(\log n + \log^3 m))$ expected time. In addition we introduce the notion of the Hilbert hull of a set of points, which we define to be the region covered by their Hilbert Delaunay triangulation. We present an algorithm for computing the Hilbert hull in time $O(nh \log^2 m)$, where h is the number of points on the hull's boundary.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Delaunay Triangulations, Hilbert metric, convexity, randomized algorithms

Digital Object Identifier 10.4230/LIPIcs.SWAT.2024.25

Related Version *Full Version*: https://arxiv.org/abs/2312.05987

1 Introduction

David Hilbert introduced the Hilbert metric in 1895 [15]. Given any convex body Ω in d-dimensional space, the Hilbert metric defines a distance between any pair of points in the interior of Ω (see Section 2 for definitions). The Hilbert metric has a number of useful properties. It is invariant under projective transformations, and straight lines are geodesics. When Ω is an Euclidean ball, it realizes the Cayley-Klein model of hyperbolic geometry. When Ω is a simplex, it provides a natural metric over discrete probability distributions (see, e.g., Nielsen and Sun [21, 22]). An excellent resource on Hilbert geometries is the handbook of Hilbert geometry by Papadopoulos and Troyanov [23].

The Hilbert geometry provides new insights into classical questions from convexity theory. Efficient approximation of convex bodies has a wide range of applications, including approximate nearest neighbor searching both in Euclidean space [6] and more general

metrics [1], optimal construction of ε-kernels [4], solving the closest vector problem approximately [11, 12, 19, 25], and computing approximating polytopes with low combinatorial complexity [3, 5]. These works all share one thing in common – they approximate a convex body by covering it with elements that behave much like metric balls. These covering elements go under various names: Macbeath regions, Macbeath ellipsoids, Dikin ellipsoids, and $(2, \varepsilon)$-covers. Vernicos and Walsh showed that these shapes are, up to a constant scaling factor, equivalent to Hilbert balls [2, 27]. In addition the Hilbert metric behaves nicely in the context flag approximability of convex polytopes as studied by Vernicos and Walsh [28].

Other applications of the Hilbert metric include machine learning [21], quantum information theory [24], real analysis [18], graph embeddings [22], and optimal mass transport [9]. Despite its obvious appeals, only recently has there been any work on developing classical computational geometry algorithms that operate in the Hilbert metric. Nielsen and Shao characterized balls in the Hilbert metric defined by a convex polygon with m sides [20]. Hilbert balls are convex polygons bounded by $2m$ sides. Nielsen and Shao showed that Hilbert balls can be computed in $O(m)$ time, and they developed dynamic software for generating them. Gezalyan and Mount presented an $O(mn \log n)$ time algorithm for computing the Voronoi diagram of n point sites in the Hilbert polygonal metric [13] (see Figure 1(a) and (b)). They showed that the diagram has worst-case combinatorial complexity of $O(mn)$. Bumpus et al. [8] further analyzed the properties of balls in the Hilbert metric and presented software for computing Hilbert Voronoi diagrams.

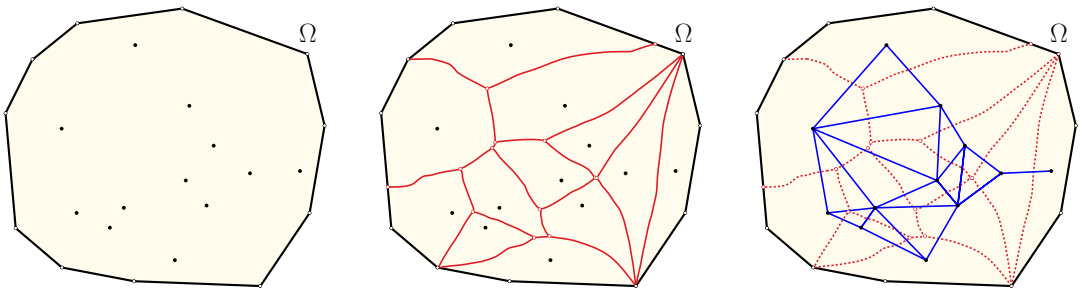

Figure 1 (a) A convex polygon Ω and sites, (b) the Voronoi diagram, and (c) the Delaunay triangulation.

In this paper, we present an algorithm for computing the Delaunay triangulation of a set P of n point sites in the Hilbert geometry defined by an m-sided convex polygon Ω. The Hilbert Delaunay triangulation is defined in the standard manner as the dual of the Hilbert Voronoi diagram (see Figure 1(c)). Our algorithm is randomized and runs in $O(n(\log n + \log^3 m))$ expected time. Excluding the polylogarithmic term in m, this matches the time of the well-known randomized algorithm for Euclidean Delaunay triangulations [14]. It is significantly more efficient than the time to compute the Hilbert Voronoi diagram, which is possible because of the smaller output size. A central element of our algorithm is an $O(\log^3 m)$ time algorithm for computing Hilbert circumcircles.

Unlike the Euclidean case, the Delaunay triangulation does not necessarily triangulate the convex hull of P. We also provide a characterization of when three points admit a Hilbert ball whose boundary contains all three points, and define the Hilbert hull, which we define to be the region covered by the Hilbert Delaunay triangulation. We give an algorithm for the Hilbert hull that runs in time $O(nh \log^2 m)$, where h is the number of points on the Hilbert hull's boundary.

2 Preliminaries

2.1 The Hilbert Metric and Hilbert Balls

The Hilbert metric is defined over the interior of a convex body Ω in $\mathbb{R}^d$ (that is, a closed, bounded, full dimensional convex set). Let $\partial\Omega$ denote Ω's boundary. Unless otherwise stated, we assume throughout that Ω is an m-sided convex polygon in $\mathbb{R}^2$. Given two points p and q in Ω, let $\overline{pq}$ denote the *chord* of Ω defined by the line through these points.

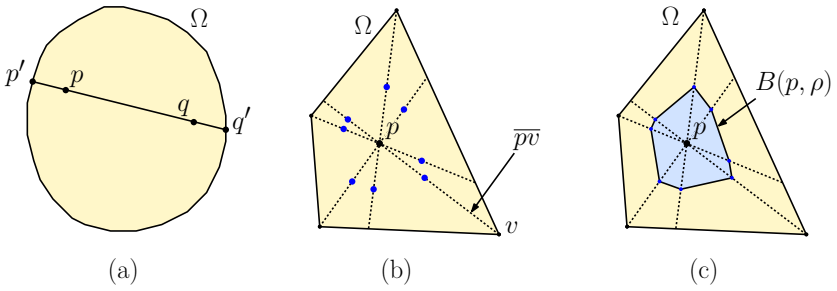

Figure 2 (a) The Hilbert distance $d_\Omega(p, q)$, (b) spokes defined by p, and (c) the Hilbert ball $B(p, \rho)$.

▶ **Definition 1** (Hilbert metric). *Given a bounded convex body Ω in $\mathbb{R}^d$ and two distinct points $p, q \in \text{int}(\Omega)$, let p' and q' denote endpoints of the chord $\overline{pq}$, so that the points are in the order $\langle p', p, q, q' \rangle$. The* Hilbert distance *between p and q is defined*

$$d_\Omega(p, q) \;=\; \frac{1}{2} \ln \left(\frac{\|q - p'\|}{\|p - p'\|} \frac{\|p - q'\|}{\|q - q'\|} \right),$$

and define $d_\Omega(p, p) = 0$ (see Figure 2(a)).

Note that the quantity in the logarithm is the cross ratio $(q, p; p', q')$. Since cross ratios are preserved by projective transformations [20], it follows that the Hilbert distances are invariant under projective transformations. Straight lines are geodesics, but not all geodesics are straight lines under the Hilbert distance. The Hilbert distance satisfies all the axioms of a metric, and in particular it is symmetric and the triangle inequality holds [23]. When Ω is a probability simplex [7], this symmetry distinguishes it from other common methods of calculating distances between probability distributions, such as the Kullback-Leibler divergence [17].

Given $p \in \text{int}(\Omega)$ and $\rho > 0$, let $B(p, \rho)$ denote the Hilbert ball of radius ρ centered at p. Nielsen and Shao showed how to compute Hilbert balls [20]. Consider the set of m chords $\overline{pv}$ for each vertex v of Ω (see Figure 2(b)). These are called the *spokes* defined by p. For each spoke $\overline{pv}$, consider the two points at Hilbert distance ρ on either side of p. $B(p, \rho)$ is the (convex) polygon defined by these $2m$ points (see Figure 2(c)). Given any line that intersects Ω, a simple binary search makes it possible to determine the two edges of Ω's boundary intersected by the line. Applying this to the line passing through p and q, it follows that Hilbert distances can be computed in $O(\log m)$ time.

2.2 The Hilbert Voronoi Diagram

Given a set P of n point sites in $\text{int}(\Omega)$, the *Hilbert Voronoi diagram* of P is defined in the standard manner as a subdivision of $\text{int}(\Omega)$ into regions, called *Voronoi cells*, based on which site of P is closest in the Hilbert distance. It was shown by Gezalyan and Mount [13] that each Voronoi cell is star-shaped with respect to its site. Given two sites $p, q \in P$, the (p, q)-*bisector* is the set of points that are equidistant from both sites in the Hilbert metric (see Figure 3(a)).

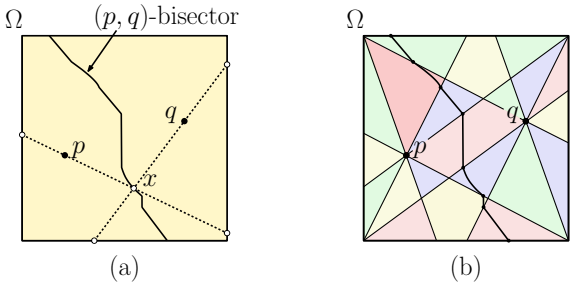

Figure 3 The (p, q)-bisector is a piecewise conic with joints on the spokes of p and q.

The distances from any point x on the (p, q)-bisector to p and q are determined by the edges of $\partial\Omega$ incident to the chords $\overline{px}$ and $\overline{qx}$. We can subdivide the interior Ω into equivalence classes based upon the identity of these edges. It is easy to see that as x crosses a spoke of either p or q, it moves into a different equivalence class. It has been shown that within each equivalence class, the bisector is a conic [8, 13]. It follows that the (p, q)-bisector is a piecewise conic, whose joints lie on sector boundaries (see Figure 3(b)). It is also known that the worst-case combinatorial complexity of the Hilbert Voronoi diagram is $\Theta(mn)$ [13].

2.3 The Hilbert Delaunay Triangulation

The *Hilbert Delaunay triangulation* of P, denoted $\text{DT}(P)$, is defined as the dual of the Hilbert Voronoi diagram, where two sites p and q are connected by an edge if their Voronoi cells are adjacent. Throughout, we make the general-position assumption that no four sites of P are Hilbert equidistant from a single point in $\text{int}(\Omega)$. It is easy to see that familiar concepts from Euclidean Delaunay triangulations apply, but using Hilbert balls rather than Euclidean balls. For example, two sites are adjacent in the triangulation if and only if there exists a Hilbert ball whose boundary contains both sites, and whose interior contains no sites. (The center of this ball lies on the Voronoi edge between the sites.) Also, three points define a triangle if and only if there is a Hilbert ball whose boundary contains all three points, but is otherwise empty.

Consider a triangle $\triangle pqr$ in Ω's interior. A Hilbert ball whose boundary passes through all three points is said to *circumscribe* this triangle. As we shall see in Section 4, some triangles admit no circumscribing Hilbert ball, but the following lemma shows that if it does exist, then it is unique. We refer to the boundary of such a ball as a *Hilbert circumcircle*. (The proof of this lemma, and a number of other lemmas throughout the paper are available in the full version of the paper.)

▶ **Lemma 2.** *There is at most one Hilbert circumcircle for any three non-colinear points in Ω's interior.*

Note that the non-collinear assumption is necessary. To see why, let Ω be a axis aligned rectangle, and let ℓ be a horizontal line through the rectangle's center. It is easy to construct two Hilbert balls, one above the line and one below, whose boundaries contain a segment along this line. Placing the three points within this segment would violate the lemma.

The following lemma shows that $\mathrm{DT}(P)$ is a connected, planar graph. Its proof is similar to the Euclidean case and appears in the full version of the paper.

▶ **Lemma 3.** *Given any discrete set of points P, $\mathrm{DT}(P)$ is a planar (straight-line) graph that spans P.*

Another useful property of the Delaunay triangulation, viewed as a graph, is its relationship to other geometric graph structures. The *Hilbert minimum spanning tree* of a point set P, denoted $\mathrm{MST}_\Omega(P)$, is the spanning tree over P where edge weights are Hilbert distances. The *Hilbert relative neighborhood graph* for a point set P, denoted $\mathrm{RNG}_\Omega(P)$ is a graph over the vertex set P, where two points $p, q \in P$ are joined by an edge if $d_\Omega(p,q) \leq \max(d_\Omega(p,r), d_\Omega(q,r))$, for all $r \in P \setminus \{p,q\}$. Toussaint proved that in the Euclidean metric, the minimum spanning tree is a subgraph of the relative neighborhood graph, which is a subgraph of the Delaunay graph [26]. The following (proved in the full version of the paper) shows that this holds in the Hilbert metric as well. (Since $\mathrm{MST}_\Omega(P)$ is connected, this provides an alternate proof that $\mathrm{DT}(P)$ spans P.)

▶ **Lemma 4.** *Given any discrete set of points P, $MST_\Omega(P) \subseteq RNG_\Omega(P) \subseteq \mathrm{DT}_\Omega(P)$.*

3 Hilbert Bisectors

In this section we present some utilities for processing Hilbert bisectors, which will be used later in our algorithms. We start by recalling a characterization given by Gezalyan and Mount for when a point lies on the Hilbert bisector [13]. Recall that three lines are said to be *concurrent* in projective geometry if they intersect in a common point or are parallel.

▶ **Lemma 5.** *Given a convex body Ω and two points $p, q \in \mathrm{int}(\Omega)$, consider any other point $x \in \Omega$. Let p' and p'' denote the endpoints of the chord $\overline{px}$ so the points appear in the order $\langle p', p, x, p'' \rangle$. Define q' and q'' analogously for $\overline{qx}$.*

- **(i)** *If $x \in \mathrm{int}(\Omega)$, then it lies on the (p,q)-bisector if and only if the lines $\overleftrightarrow{pq}$, $\overleftrightarrow{p'q'}$, and $\overleftrightarrow{p''q''}$ are concurrent (see Figure 4(a)).*
- **(ii)** *If $x \in \partial\Omega$ (implying that $x = p'' = q''$), then x is the limit point of the (p,q)-bisector if and only if there is a supporting line through x concurrent with $\overleftrightarrow{pq}$ and $\overleftrightarrow{p'q'}$ (see Figure 4(b)).*

The following utility lemmas arise as consequences of this characterization. Both involve variants of binary search. Their proofs are given in the full version of the paper.

▶ **Lemma 6.** *Given an m-sided convex polygon Ω, two points $p, q \in \mathrm{int}(\Omega)$, and any ray emanating from p, the point of intersection between the ray and the (p,q)-bisector (if it exists) can be computed in $O(\log^2 m)$ time.*

▶ **Lemma 7.** *Given an m-sided convex polygon Ω and any two points $p, q \in \mathrm{int}(\Omega)$, the endpoints of the (p,q)-bisector on $\partial\Omega$ can be computed in $O(\log^2 m)$ time.*

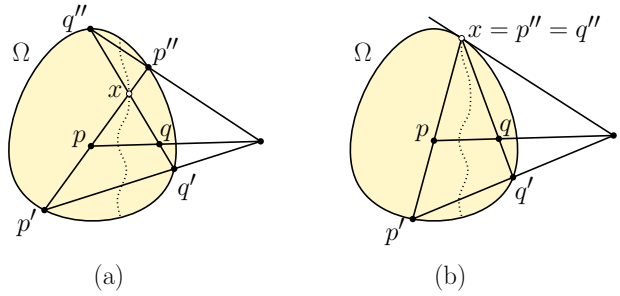

Figure 4 Properties of points on the Hilbert bisector.

4 Hilbert Circumcircles

In the Euclidean plane, any triple of points that are not collinear lie on a unique circle, that is, the boundary of an Euclidean ball. This is not true in the Hilbert geometry, however. Since Delaunay triangulations are based on an empty circumcircle condition, it will be important to characterize when a triangle admits a Hilbert circumcircle and when it does not. In this section we explore the conditions under which such a ball exists.

4.1 Balls at infinity

We begin by introducing the concept of a Hilbert ball centered at a point on the boundary of Ω. Let x be any point on $\partial\Omega$. Given any point $p \in \text{int}(\Omega)$, we are interested in defining the notion of a Hilbert ball centered at x whose boundary contains p. If we think of the points on the boundary of Ω as being infinitely far from any point in $\text{int}(\Omega)$, this gives rise to the notion of a ball at infinity.

Given $x \in \partial\Omega$ and $p \in \text{int}(\Omega)$, let u be any unit vector directed from x into the interior of Ω (see Figure 5(a)). Given any sufficiently small positive δ, let $x_\delta = x + \delta u$ denote the point at Euclidean distance δ from x along vector u, and let $B(x_\delta, p)$ denote the Hilbert ball centered at x_δ of radius $d_\Omega(x_\delta, p)$. The following lemma shows that as δ approaches 0, this ball approaches a shape, which we call the *Hilbert ball at infinity* determined by x and u and passing through p, denoted $B_u(x, p)$ (see Figure 5(b)).

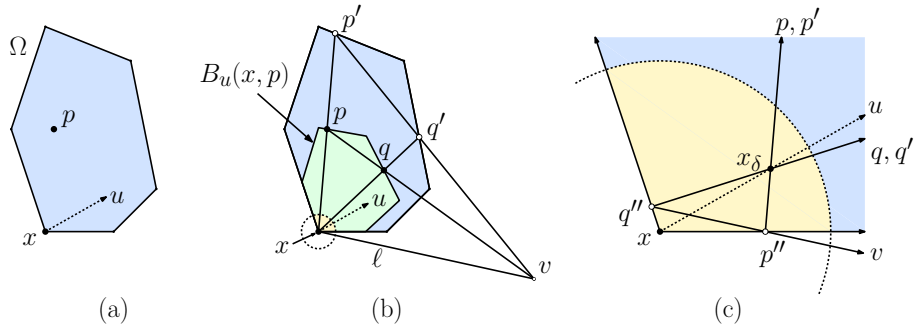

Figure 5 Constructing the Hilbert ball $B_u(x, p)$ at infinity.

▶ **Lemma 8.** *As δ approaches 0, $B(x_\delta, p)$ approaches a convex polygon lying within Ω having x on its boundary.*

A useful utility, which we will apply in Section 7, involves computing the largest empty ball centered at any boundary point x with respect to a point set P. The proof is given in the full version of the paper.

▶ **Lemma 9.** *Given a set of n points P in the interior of Ω, and any point $x \in \partial\Omega$, in $O(n \log m)$ time, it is possible to compute a point $p \in P$, such that there is a ball at infinity centered at x whose boundary passes through p, and which contains no points of P in its interior.*

Balls at infinity are not proper aspects of Hilbert geometry, but they will be convenient for our purposes. Given two points $p, q \in \text{int}(\Omega)$, let x denote the endpoint of the (p, q)-bisector on $\partial\Omega$, oriented so that x lies to the left of the directed line $\overrightarrow{pq}$. (It follows from the star-shapedness of Voronoi cells that the bisector endpoints intersect $\partial\Omega$ on opposite sides of this line.) Let u denote a tangent (if x is a vertex of Ω) vector of the bisector at x. Define $B(p:q) = B_u(x, p)$. Note that this (improper) ball is both centered at and passes through x. In this sense it circumscribes the triangle $\triangle pqx$. Define $B(q:p)$ analogously for the opposite endpoint of this bisector (see Figure 6(a) and (b)).

We can now characterize the set of points r that admit a Hilbert circumcircle with respect to two given points p and q. This characterization is based on two regions, called the *overlap* and *outer regions* (see Figure 6(c)).

▶ **Definition 10** (Overlap/Outer Regions). *Given two points $p, q \in \text{int}(\Omega)$:*

Overlap Region: *denoted $Z(p, q)$, is $B(p:q) \cap B(q:p)$.*
Outer Region: *denoted $W(p, q)$, is $\Omega \setminus (B(p:q) \cup B(q:p))$.*

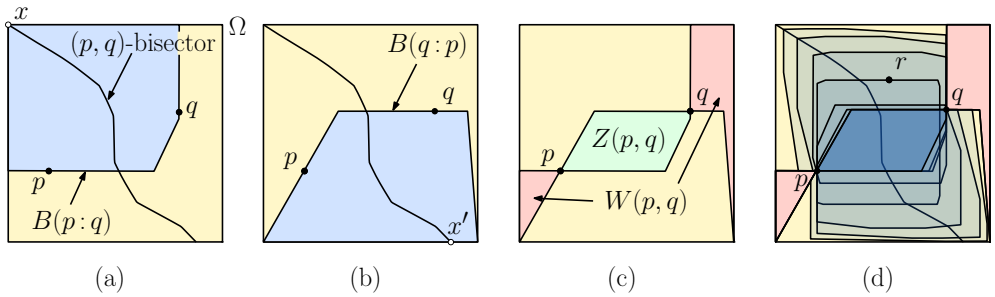

Figure 6 The overlap region $Z(p, q)$ and the outer region $W(p, q)$.

▶ **Lemma 11.** *A triangle $\triangle pqr \subseteq \text{int}(\Omega)$ admits a Hilbert circumcircle if and only if $r \notin Z(p, q) \cup W(p, q)$.*

As shown in Figure 1, the Delaunay triangulation need not cover the convex hull of the set of sites. We refer to the region that is covered as the *Hilbert hull* of the sites. Later, we will present an algorithm for computing the Hilbert hull. The following lemma will be helpful. It establishes a nesting property for the overlap regions.

▶ **Lemma 12.** *If $r \in Z(p, q)$ then $Z(p, r) \subset Z(p, q)$.*

5 Computing Circumcircles

A fundamental primitive in the Euclidean Delaunay triangulation algorithm is the so-called *in-circle test* [14]. Given a triangle $\triangle pqr$ and a fourth site s, the test determines whether s lies within the circumscribing Hilbert ball for the triangle (if such a ball exists). In this

section, we present an algorithm which given any three sites either computes the Hilbert circumcircle for these sites, denoted $B(p:q:r)$, or reports that no circumcircle exists. The in-circle test reduces to checking whether $d_\Omega(s,c) < \rho$, which can be done in $O(\log m)$ time as observed in Section 2.

▶ **Lemma 13.** *Given a convex m-sided polygon Ω and triangle $\triangle pqr \subset \mathrm{int}(\Omega)$, in $O(\log^3 m)$ time it is possible to compute $B(p:q:r)$ or to report that no ball exists.*

The remainder of the section is devoted to the proof. The circumscribing ball exists if and only if there is a point equidistant to all three, implying that (p,q)- and (p,r)-bisectors intersect at some point $c \in \mathrm{int}(\Omega)$. The following technical lemma shows that bisectors intersect crosswise.

▶ **Lemma 14.** *Given three non-collinear points $p,q,r \in \mathrm{int}(\Omega)$, the (p,q)- and (p,r)-bisectors have endpoints lying on $\partial\Omega$. If they intersect within $\mathrm{int}(\Omega)$, they intersect transversely in a single point (see Figure 7).*

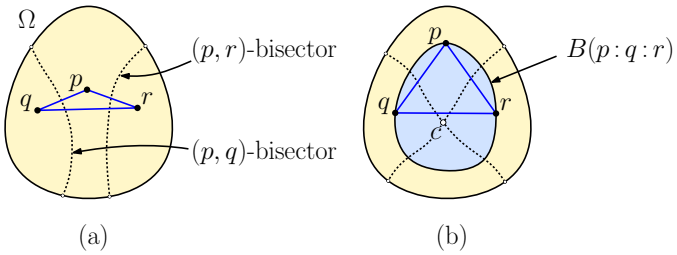

Figure 7 Circumcircles and bisector intersection.

It follows that the (p,q)- and (p,r)-bisectors subdivide the interior of Ω into either three or four regions (three if they do not intersect, and four if they do). We can determine which is this case by invoking Lemma 7 to compute the endpoints of these bisectors in $O(\log^2 m)$ time. If they alternate between (p,q) and (p,r) along the boundary of Ω, then the bisectors intersect, and otherwise they do not. In the latter case, there is no circumcircle for p, q, and r, and hence, no possibility of violating the circumcircle condition. (Note that this effectively provides an $O(\log^2 m)$ test for Lemma 11.) Henceforth, we concentrate on the former case.

Let us assume, without loss of generality, that $\triangle pqr$ is oriented counterclockwise. Let v_q denote the endpoint of the (p,q)-bisector that lies to the right of the oriented line $\overrightarrow{pq}$. Let v_r denote the endpoint of the (p,r)-bisector that lies to the left of the oriented line $\overrightarrow{pr}$ (see Figure 8(a)).

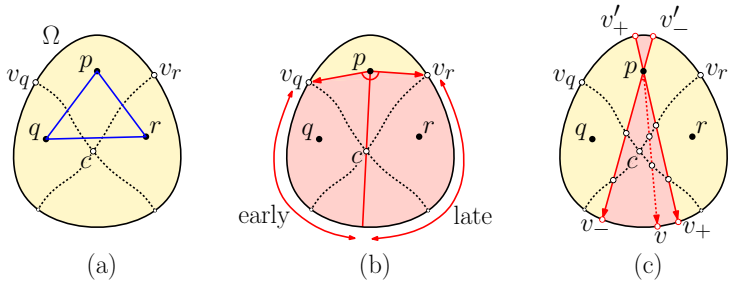

Figure 8 Computing the center of $\triangle pqr$.

Because Voronoi cells are star-shaped, it follows that the vector from p to the desired circumcenter c lies in the counterclockwise angular interval from $\overrightarrow{pv_q}$ to $\overrightarrow{pv_r}$ (see Figure 8(b)). The key to the search is the following observation. In the angular region from $\overrightarrow{pv_q}$ to $\overrightarrow{pc}$, any ray shot from p intersects the (p,q)-bisector before hitting the (p,r)-bisector (if it hits the (p,r)-bisector at all). On the other hand, in the angular region from $\overrightarrow{pc}$ to $\overrightarrow{pv_r}$, any ray shot from p intersects the (p,r)-bisector before hitting the (p,q)-bisector (if it hits the (p,q)-bisector at all). We say that the former type of ray is *early* and the latter type is *late*.

Let v_- and v_+ denote the points on $\partial\Omega$ that bound the current search interval about p (see Figure 8(c)). We will maintain the invariant that the ray $\overrightarrow{pv_-}$ is early and $\overrightarrow{pv_+}$ is late. Initially, $v_- = v_q$ and $v_+ = v_r$. Let v'_- and v'_+ denote the opposite endpoints of the chords $\overline{pv_-}$ and $\overline{pv_+}$. Consider the portion of the boundary of Ω that lies counterclockwise from v_- and v_+. If the angle $\angle v_- p v_+$ is smaller than π, also consider the portion of the boundary of Ω that lies counterclockwise from v'_- and v'_+. With each probe, we sample the median vertex v from whichever of these two boundary portions that contains the larger number of vertices. If v comes from the interval $[v_-, v_+]$, we probe along the ray $\overrightarrow{pv}$, and if it comes from the complementary interval, $[v'_-, v'_+]$, we shoot the ray from p in the opposite direction from v. We then apply Lemma 6 twice to determine in $O(\log^2 m)$ time where this ray hits the (p,q)- and (p,r)-bisectors (if at all). Based on the results, we classify this ray as being early or late and recurse on the appropriate angular subinterval. When the search terminates, we have determined a pair of consecutive spokes about p that contain c.

Because each probe eliminates at least half of the vertices from the larger of the two boundary portions, it follows that at $O(\log m)$ probes, we have located c to within a single pair of consecutive spokes around p. Since each probe takes $O(\log^2 m)$ time, the entire search takes $O(\log^3 m)$ time.

We repeat this process again for r and q. The result is three double wedges defined by consecutive spokes, one about each site. It follows from our earlier remarks from Section 2.2 that within the intersection of these regions, the bisectors are simple conics. We can compute these conics in $O(1)$ time [8] and determine their intersection point, thus yielding the desired point c. The radius ρ of the ball can also be computed in $O(1)$ time.

6 Building the Triangulation

In this section we present our main result, a randomized incremental algorithm for constructing the Delaunay triangulation $\mathrm{DT}(P)$ for a set of n sites P in the interior of an m-sided convex polygon Ω. Our algorithm is loosely based on a well-known randomized incremental algorithm for the Euclidean case [10, 14].

6.1 Orienting and Augmenting the Triangulation

In this section we introduce some representational conventions for the sake of our algorithm. First, we will orient the elements of the triangulation. Given $p, q \in \mathrm{int}(\Omega)$ define the *endpoint* of the (p,q)-bisector to be the endpoint that lies to the left of the directed line $\overrightarrow{pq}$. (It is worth repeating that it follows from the star-shapedness of Voronoi cells that the bisector endpoints intersect $\partial\Omega$ on opposite sides of this line.) The opposite endpoint will be referred to as the (q,p)-bisector endpoint. When referring to a triangle $\triangle pqr$, we will assume that the vertices are given in counterclockwise order. Also, edges in the triangulation are assumed to be directed, so we can unambiguously reference the left and right sides of a directed edge pq.

As mentioned earlier, the triangulation need not cover the convex hull of the set of sites (see Figure 9(a)). For the sake of construction, it will be convenient to augment the triangulation with additional elements, so that all of Ω is covered. First, we add elements

to include the endpoints of the Voronoi bisectors on $\partial\Omega$. For each edge pq such that the external face of the triangulation lies to its left, the endpoint of the (p, q)-bisector intersects the boundary of Ω. Letting x denote this boundary point, we add a new triangle $\triangle pqx$, called a *tooth*. (See the green shaded triangles in Figure 9(b).)

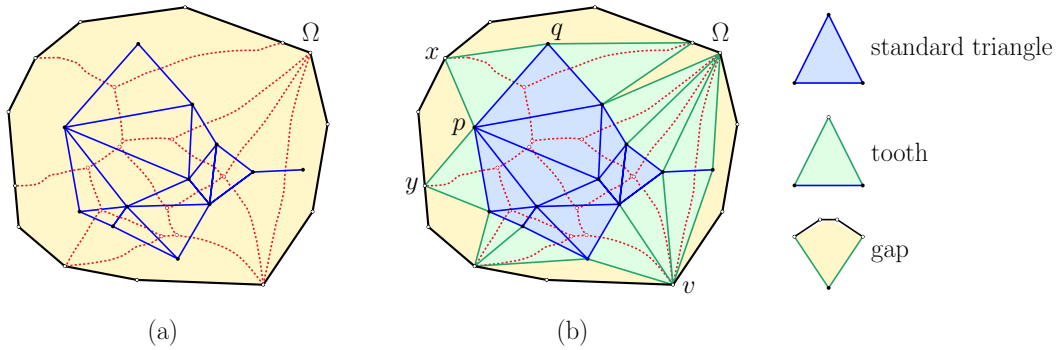

Figure 9 The augmented triangulation.

Finally, the portion of Ω that lies outside of all the standard triangles and teeth consists of a collection of regions, called *gaps*, each of which involves a single site, the sides of two teeth, and a convex polygonal chain along $\partial\Omega$. While these shapes are not triangles, they are defined by three points, and we will abuse the notation $\triangle pxy$ to refer to the gap defined by the site p and boundary points x and y (see Figure 9(b)).

Even when sites are in general position, multiple teeth can share the same boundary vertex. For example, in Figure 9(b) three teeth meet at the same boundary vertex v. In our augmented representation, it will be convenient to treat these as three separate teeth, meeting at three distinct (co-located) vertices, separated by two degenerate (zero-width) gaps. This allows us to conceptualize the region outside of the standard triangulation as consisting of an alternating sequence of teeth and gaps. The following lemma summarizes a few useful facts about the augmented representation. The proof is straightforward and appears in the full version of the paper.

▶ **Lemma 15.** *Given a set of n point sites in the interior of a convex polygon Ω:*

- *The augmented Delaunay triangulation has complexity $O(n)$.*
- *The region covered by standard triangles is connected.*
- *Each standard triangle and each tooth satisfies the empty circumcircle property.*
- *The region outside the standard triangles consists of an alternating sequence of teeth and gaps.*
- *For any $p \in \Omega$, membership in any given triangle, tooth, or gap can be determined in $O(1)$ time.*

6.2 Local and Global Delaunay

Given a set P of point sites, we say that an augmented triangulation $\mathcal{T}(P)$ is *globally Delaunay* (or simply *Delaunay*) if for each standard triangle and each tooth $\triangle pqr$, the circumscribing ball, denoted $B(p:q:r)$, exists and has no site within its interior. The incremental Euclidean Delaunay triangulation algorithm employs a local condition, which only checks this for neighboring triangles [10, 14]. Given a directed edge pq of the triangulation that is incident to two triangles or to a triangle and tooth, let a and b be the vertices of the incident triangles lying to the left and right of the pq, respectively. We say that $\mathcal{T}(P)$ is *locally Delaunay* if

$a \notin \text{int}(B(q:p:b))$ and $b \notin \text{int}(B(p:q:a))$ for all such edges. The following lemma shows that it suffices to certify the Delaunay properties locally. The proof, which appears in the full version of the paper, follows the same structure as the Euclidean case, but additional care is needed due to the existence of teeth and gaps.

▶ **Lemma 16.** *Given a set P of point sites, an augmented triangulation $\mathcal{T}(P)$ is globally Delaunay if and only if it is locally Delaunay.*

6.3 Incremental Construction

The algorithm operates by first randomly permuting the sites. It begins by inserting two arbitrary sites a and b and generating the resulting augmented triangulation. In $O(\log^2 m)$ time, we can compute the endpoints of the (a, b)-bisector. We add the edge ab and create two teeth by connecting a and b to the bisector endpoints (see Figure 10(a)). We then insert the remaining points one by one, updating the triangulation incrementally after each insertion (see Algorithm 1). Later, we will discuss how to determine which element each new site lies in, but for now, let us focus on how the triangulation is updated.

Algorithm 1 Constructs the Hilbert Delaunay triangulation of a point set P.

```
procedure DELAUNAY(P)                    ▷ Build the Delaunay triangulation of point set P
    T ← empty triangulation
    Randomly permute P
    a, b ← any two points of P           ▷ Initialize with sites a and b
    x, y ← endpoints of the (a, b)-bisector   ▷ See Figure 10(a)
    Add edges ab, ax, ay, bx, by to T
    for all p ∈ P \ {a, b} do            ▷ Add all remaining sites
        INSERT(p, T)
    end for
    return T
end procedure
```

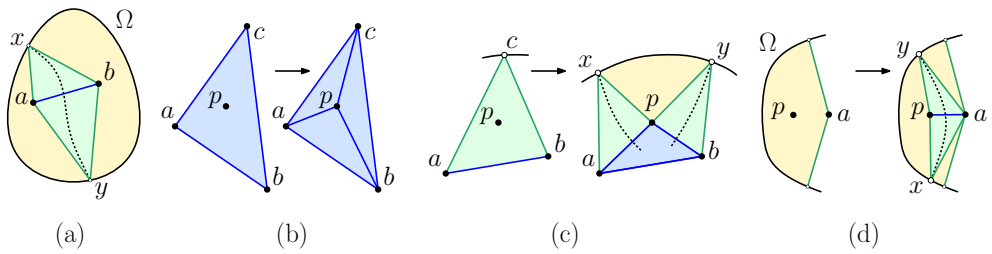

Figure 10 (a) Initialization and insertion into a (b) standard triangle, (c) tooth, (d) gap.

When we insert a point, there are three possible cases, depending on the type of element that contains the point, a standard triangle (three sites), a tooth (two sites), or a gap (one site). The case for standard triangles is the same as for Euclidean Delaunay triangulations [14], involving connecting the new site to the triangle's vertices (see Figure 10(b)). Insertion into a tooth involves removing the boundary vertex, connecting the new site to the two existing site vertices, and creating two new teeth based on the bisectors to these sites (see Figure 10(c)). Finally, insertion into a gap involves connecting the new site to the existing site vertex, and creating two new teeth based on the bisectors to this site (see Figure 10(d)).

Algorithm 2 Site insertion.

```
procedure INSERT(p, T)                          ▷ Insert a new site p into triangulation T
    △abc ← the triangle of T containing p
    if △abc is a standard triangle then                                  ▷ See Figure 10(b)
        Add edges ap, bp, cp to T
        FLIPEDGE(ab, p, T); FLIPEDGE(bc, p, T); FLIPEDGE(ca, p, T)
    else if △abc is a tooth then                                         ▷ See Figure 10(c)
        Let a and b be sites, and c be on boundary
        x, y ← endpoints of the (a, p)- and (p, b)-bisectors, respectively
        Remove c and edges ca and cb from T
        Add edges ap, bp, ax, px, by, py to T
        FLIPEDGE(ab, p, T);
        FIXTOOTH(△apx, p, T); FIXTOOTH(△pby, p, T)
    else                                          ▷ △abc is a gap; see Figure 10(d)
        Let a be the site, and let b and c be on the boundary
        x, y ← endpoints of (a, p)- and (p, a)-bisectors, respectively
        Add edges ap, ax, px, ay, py to T
        FIXTOOTH(△apx, p, T); FIXTOOTH(△pay, p, T)
    end if
end procedure
```

On return from INSERT, $\mathcal{T}$ is a topologically valid augmented triangulation, but it may fail to satisfy the Delaunay empty circumcircle conditions, and may even fail to be geometrically valid. The procedures FLIPEDGE and FIXTOOTH repair any potential violations of the local Delaunay conditions.

The procedure FLIPEDGE is applied to all newly generated standard triangles $\triangle pab$. It is given the directed edge ab of the triangle, such that p lies to the left of this edge. It accesses the triangle $\triangle abc$ lying to the edge's right. This may be a standard triangle or a tooth. In either case, it has an associated circumcircle, which we assume has already been computed.[1] We test whether p encroaches on this circumcircle, and if so, we remove the edge ab. If $\triangle abc$ is a standard triangle, then we complete the edge-flip by adding edge pc, and then continue by checking the edges ac and cb (see Figure 11(a)). Otherwise, we remove vertex c, and compute the endpoints x and y of the (p, a)- and (b, p)-bisectors, respectively. We create two new teeth, $\triangle pax$ and $\triangle bpy$ (see Figure 11(b)). This creates a new (possibly degenerate) gap $\triangle pxy$. Later, we will show (see Lemma 17) that no further updates are needed. The algorithm is presented in Algorithm 3.

The "else" clause creates two teeth $\triangle pax$ and $\triangle bpy$. The following lemma (proved in the full version of the paper) shows that there is no need to apply FIXTOOTH to them.

▶ **Lemma 17.** *On return from FLIPEDGE the teeth $\triangle pax$ and $\triangle bpy$ generated in the "else" clause are both valid.*

Finally, we present FIXTOOTH. To understand the issue involved, consider Figure 10(d). In the figure, the newly created teeth $\triangle pax$ and $\triangle apy$ both lie entirely inside the existing gap. But, this need not generally be the case, and the newly created boundary vertex may

[1] In the Euclidean case, the in-circle test may be run from either $\triangle abc$ or $\triangle pab$, but this is not true for the Hilbert geometry. In light of Lemma 11, we do not know whether the $\triangle pab$ has a circumcircle, so we run the test from $\triangle abc$.

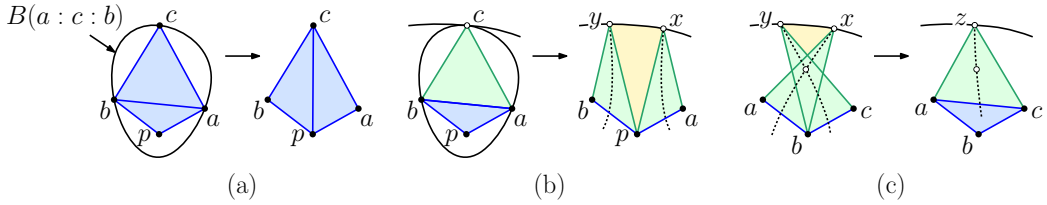

Figure 11 (a) Standard-triangle edge flip, (b) tooth edge flip, and (c) fixing a tooth.

Algorithm 3 In-circle test and edge flip.

procedure FLIPEDGE($ab, p, \mathcal{T}$) ▷ In-circle test. New site p lies to the left of edge ab.
  $c \leftarrow$ vertex of triangle to right of ab in $\mathcal{T}$
  if $p \in B(a:c:b)$ **then** ▷ p fails the in-circle test for $\triangle acb$
    Remove edge ab from $\mathcal{T}$
    if $\triangle acb$ is a standard triangle **then** ▷ See Figure 11(a)
      Add edge pc to $\mathcal{T}$
      FLIPEDGE($ac, p, \mathcal{T}$); FLIPEDGE($cb, p, \mathcal{T}$) ▷ Create $\triangle pac$ and $\triangle bpc$
    else ▷ $\triangle acb$ is a tooth. See Figure 11(b)
      $x, y \leftarrow$ endpoints of (p, a)- and (b, p)-bisectors, respectively
      Remove c and edges cb and ca from $\mathcal{T}$
      Add edges ax, px, by, py to $\mathcal{T}$ ▷ Create teeth $\triangle pax$ and $\triangle bpy$
    end if
  end if
end procedure

lie outside the current gap, resulting in two or more teeth that overlap each other (see, e.g., Figure 11(c)). The procedure FIXTOOTH is given a tooth $\triangle abx$, where a and b are sites, and x is on Ω's boundary. Whenever it is invoked the new site p is either a or b. Recalling our assumption that teeth and gaps alternate around the boundary of Ω, we check the teeth that are expected to be adjacent to the current tooth on the clockwise and counterclockwise sides. (Only the clockwise case is presented in the algorithm, but the other case is symmetrical, with a and b swapped.)

Let $\triangle bcy$ denote the tooth immediately clockwise around the boundary. We test whether these triangles overlap (see Figure 11(c)). If so, we know that the (a, b)-bisector (which ends at x) and the (b, c)-bisector (which ends at y) must intersect. This intersection point is the center of a Hilbert ball circumscribing $\triangle abc$. We replace the two teeth $\triangle abx$ and $\triangle bcy$ with the standard triangle $\triangle abc$ and the tooth $\triangle acz$, where z is the endpoint of the (a, c)-bisector. If $p = a$, then we invoke FLIPEDGE on the opposite edge bc from p, and we invoke FIXTOOTH on the newly created tooth. When the algorithm terminates, all the newly generated elements have been locally validated. The algorithm is presented in Algorithm 4.

Based on our earlier remarks, it follows that our algorithm correctly inserts a site into the augmented triangulation.

▶ **Lemma 18.** *Given an augmented triangulation $\mathcal{T}(P)$ for a set of sites P, the procedure* INSERT *correctly inserts a new site p, resulting in the augmented Delaunay triangulation for of $P \cup \{p\}$.*

The final issue is the algorithm's expected running time. Our analysis follows directly from the analysis of the randomized incremental algorithm for the Euclidean Delaunay triangulation. We can determine which triangle contains each newly inserted point in

Algorithm 4 Check for and fix overlapping teeth.

```
procedure FixTooth(△abx, p, T)          ▷ Fix tooth △abx where p = a or p = b
    Let △bcy be the tooth clockwise adjacent to △abx
    if △abx overlaps △bcy then                                  ▷ See Figure 11(c)
        z ← endpoint of the (a, c)-bisector
        Remove x and y and edges ax, bx, by, and cy from T
        Add edges ac, az, and cz to T          ▷ Create triangle △abc and tooth △acz
        if p = a then
            FlipEdge(bc, p, T)          ▷ Check edge flip with triangle opposite bc
            FixTooth(△acz, p, T)                          ▷ Check for further overlaps
        end if
    else
        Repeat the above for the triangle counterclockwise from △abx swapping a ↔ b
    end if
end procedure
```

amortized $O(\log n)$ expected time either by building a history-based point location data structure (as with Guibas, Knuth, and Sharir [14]) or by bucketing sites (as with de Berg et al. [10]). The analysis in the Euclidean case is based on a small number of key facts, which apply in our context as well. First, sites are inserted in random order. Second, the conflict set for any triangle or tooth consists of the points lying in the triangle's circumcircle. Both of these clearly hold in the Hilbert setting. Third, the number of structural updates induced by the insertion of site p is proportional to the degree of p following the insertion. This holds for our algorithm, because each modification to the triangulation induced by p's insertion results in a new edge being added to p (and these edges are not deleted until future insertions). Fourth, the structure is invariant to the insertion order. Finally, the triangulation graph is planar, and hence it has constant average degree. The principal difference is that the in-circle test involves computing a Hilbert circumcircle, which by Lemma 13 can be performed in $O(\log^3 m)$ time. These additional factors of $O(\log n)$ and $O(\log^3 m)$ are performed a constant number of times in expectation, for each of the n insertions. This implies our main result.

▶ **Theorem 19.** *Given a set of n points in the Hilbert geometry defined by a convex m-gon Ω, it is possible to construct the augmented Delaunay triangulation in randomized expected time $O(n(\log n + \log^3 m))$.*

7 The Hilbert Hull

As observed earlier, the triangles of the Delaunay triangulation of P do not necessarily cover the convex hull of P. The region covered by these triangles is called the *Hilbert hull* (the blue region of Figure 9). In this section, we present a simple algorithm for computing this hull for a set of n points in the Hilbert distance defined by a convex m-gon Ω as an alternative to deriving it from the Delaunay triangulation of the point-set. Our approach is roughly based on the Jarvis march algorithm [16] for computing convex hulls.

The standard Jarvis march maintains two consecutive sites on the hull, and it iteratively computes the next point in $O(n)$ time using an angular ordering induced by these two points. Our algorithm will instead maintain a current site p on the hull, and a boundary point x, such that there is an empty ball at infinity centered at x whose boundary passes through p. It uses x and p to induce an angular order to select the next point.

To start the process off, we need to identify a pair (p_0, x_0), where $p_0 \in P$, $x_0 \in \partial\Omega$, and p_0 lies on the boundary of an empty ball centered at x_0. By Lemma 9, such a pair can be computed in $O(n \log m)$ time. Assuming that the algorithm has generated an initial sequence of points on the hull, arriving at a pair (p_i, x_i) satisfying the above invariant, we compute the next pair as follows. First, we take r to be the point of P that minimizes the counterclockwise angle $\angle x_i p_i r$. Such a point has not already been added to the hull.

We then enumerate the points of $P \setminus \{p_i, r\}$ to see whether any lie in the overlap region $Z(p, r)$. Whenever we encounter such a point, we set r to this point and continue the process. After considering all the points of P, it follows from nesting properties of overlap regions (see Lemma 12) that $Z(p, r)$ is empty, and hence the (r, p)-bisector extends to the boundary of Ω in the Voronoi diagram of P. Thus, we take $p_{i+1} \leftarrow r$, and x_{i+1} is the bisector endpoint. From the remarks made after Lemma 14, we can test membership in the $Z(p, r)$ in $O(\log^2 m)$ time, and by Lemma 7, we can compute the endpoint on the bisector in $O(\log^2 m)$ time as well. Since we consider at most n points, it takes a total of $O(n \log^2 m)$ time to generate one more point on the Hilbert hull. This implies that the overall running time is $O(nh \log^2 m)$.

▶ **Lemma 20.** *Given a set of n points in the Hilbert geometry defined by a convex m-gon Ω, it is possible to construct the Hilbert hull in time $O(nh \log^2 m)$, where h is the number of points on the hull's boundary.*

8 Concluding Remarks

In this paper we presented an algorithm for computing the Delaunay triangulation of a set P of n point sites in the Hilbert geometry defined by an m-sided convex polygon Ω in expected $O(n(\log n + \log^3 m)$ time. Additionally we presented an algorithm for the Hilbert Hull, the boundary of the Hilbert Delaunay triangulation, in time $O(nh \log^2 m)$, where h is the number of points on the boundary. Supporting results, such as the algorithm for determining if it exists and computing the circumscribing ball of three non-collinear points in the interior of Ω in $O(\log^3 m)$ time, and the characterization of Hilbert balls at infinity, may be useful for further algorithmic results in the Hilbert metric. As discussed in the introduction, the Hilbert metric has a large variety of applications including convex approximation [2] [28], machine learning [21], quantum information theory [24], real analysis [18], graph embeddings [22], and optimal mass transport [9]. Extensions of algorithmic results from Euclidean to Hilbert geometry may prove of use to researchers in these fields.

References

1 Ahmed Abdelkader, Sunil Arya, Guilherme Dias da Fonseca, and David M. Mount. Approximate nearest neighbor searching with non-Euclidean and weighted distances. In *Proc. 30th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 355–372, 2019. doi:10.1137/1.9781611975482.23.

2 Ahmed Abdelkader and David M. Mount. Economical Delone sets for approximating convex bodies. In *Proc. 16th Scand. Workshop Algorithm Theory*, pages 4:1–4:12, 2018. doi:10.4230/LIPIcs.SWAT.2018.4.

3 Rahul Arya, Sunil Arya, Guilherme Dias da Fonseca, and David M. Mount. Optimal bound on the combinatorial complexity of approximating polytopes. *ACM Trans. Algorithms*, 18:1–29, 2022. doi:10.1145/3559106.

4 Sunil Arya, Guilherme Dias da Fonseca, and David M. Mount. Near-optimal ε-kernel construction and related problems. In *Proc. 33rd Internat. Sympos. Comput. Geom.*, pages 10:1–15, 2017. doi:10.4230/LIPIcs.SoCG.2017.10.

5 Sunil Arya, Guilherme Dias da Fonseca, and David M. Mount. On the combinatorial complexity of approximating polytopes. *Discrete Comput. Geom.*, 58(4):849–870, 2017. doi:10.1007/s00454-016-9856-5.

6 Sunil Arya, Guilherme Dias da Fonseca, and David M. Mount. Approximate polytope membership queries. *SIAM J. Comput.*, 47(1):1–51, 2018. doi:10.1137/16M1061096.

7 Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. doi:10.1017/CBO9780511804441.

8 Madeline Bumpus, Caesar Dai, Auguste H. Gezalyan, Sam Munoz, Renita Santhoshkumar, Songyu Ye, and David M. Mount. Software and analysis for dynamic Voronoi diagrams in the Hilbert metric, 2023. arXiv:2304.02745.

9 Yongxin Chen, Tryphon Georgiou, and Michele Pavon. Entropic and displacement interpolation: A computational approach using the Hilbert metric. *SIAM J. Appl. Math.*, 76:2375–2396, 2016. doi:10.1137/16M1061382.

10 Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2010. doi:10.1007/978-3-540-77974-2.

11 Friedrich Eisenbrand, Nicolai Hähnle, and Martin Niemeier. Covering cubes and the closest vector problem. In *Proc. 27th Annu. Sympos. Comput. Geom.*, pages 417–423, 2011. doi:10.1145/1998196.1998264.

12 Friedrich Eisenbrand and Moritz Venzin. Approximate CVPs in time $2^{0.802n}$. *J. Comput. Sys. Sci.*, 124:129–139, 2021. doi:10.1016/j.jcss.2021.09.006.

13 Auguste H Gezalyan and David M Mount. Voronoi diagrams in the hilbert metric. In *39th International Symposium on Computational Geometry (SoCG 2023)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2023.

14 Leonidas J. Guibas, Donald E. Knuth, and Micha Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7:381–413, 1992. doi:10.1007/BF01758770.

15 D. Hilbert. Ueber die gerade Linie als kürzeste Verbindung zweier Punkte. *Math. Annalen*, 46:91–96, 1895. doi:10.1007/BF02096204.

16 Ray A Jarvis. On the identification of the convex hull of a finite set of points in the plane. *Information processing letters*, 2(1):18–21, 1973.

17 S. Kullback and R. A. Leibler. On information and sufficiency. *Annals. Math. Stat.*, 22:79–86, 1951. doi:10.1214/aoms/1177729694.

18 Bas Lemmens and Roger Nussbaum. Birkhoff's version of Hilbert's metric and its applications in analysis, 2013. arXiv:1304.7921.

19 Márton Naszódi and Moritz Venzin. Covering convex bodies and the closest vector problem. *Discrete Comput. Geom.*, 67:1191–1210, 2022. doi:10.1007/s00454-022-00392-x.

20 Frank Nielsen and Laetitia Shao. On balls in a Hilbert polygonal geometry. In *Proc. 33rd Internat. Sympos. Comput. Geom.*, pages 67:1–67:4, 2017. (Multimedia contribution). doi:10.4230/LIPIcs.SoCG.2017.67.

21 Frank Nielsen and Ke Sun. Clustering in Hilbert's projective geometry: The case studies of the probability simplex and the elliptope of correlation matrices. In Frank Nielsen, editor, *Geometric Structures of Information*, pages 297–331. Springer Internat. Pub., 2019. doi:10.1007/978-3-030-02520-5_11.

22 Frank Nielsen and Ke Sun. Non-linear embeddings in Hilbert simplex geometry, 2022. arXiv:2203.11434.

23 Athanase Papadopoulos and Marc Troyanov. *Handbook of Hilbert geometry*, volume 22 of *IRMA Lectures in Mathematics and Theoretical Physics*. European Mathematical Society Publishing House, 2014. doi:10.4171/147.

24 David Reeb, Michael J. Kastoryano, and Michael M. Wolf. Hilbert's projective metric in quantum information theory. *J. Math. Physics*, 52(8), 2011. doi:10.1063/1.3615729.

25 Thomas Rothvoss and Moritz Venzin. Approximate CVP in time $2^{0.802n}$ – Now in any norm! In *Proc. 23rd Internat. Conf. on Integ. Prog. and Comb. Opt. (IPCO 2022)*, pages 440–453, 2022. doi:10.1007/978-3-031-06901-7_33.

26 Godfried T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recogn.*, 12:261–268, 1980. doi:10.1016/0031-3203(80)90066-7.

27 Constantin Vernicos. On the Hilbert geometry of convex polytopes. In *Handbook of Hilbert geometry*, volume 22 of *IRMA Lectures in Mathematics and Theoretical Physics*, pages 111–126. European Mathematical Society Publishing House, 2014. doi:10.48550/arXiv.1406.0733.

28 Constantin Vernicos and Cormac Walsh. Flag-approximability of convex bodies and volume growth of Hilbert geometries, 2018. arXiv:1809.09471.