The Complexity of Geodesic Spanners

Sarita de Berg
Department of Information and Computing Sciences, Utrecht University, The Netherlands

Marc van Kreveld
Department of Information and Computing Sciences, Utrecht University, The Netherlands

Frank Staals
Department of Information and Computing Sciences, Utrecht University, The Netherlands

Abstract

A *geometric t-spanner* for a set S of n point sites is an edge-weighted graph for which the (weighted) distance between any two sites $p, q \in S$ is at most t times the original distance between p and q. We study geometric t-spanners for point sets in a constrained two-dimensional environment P. In such cases, the edges of the spanner may have non-constant complexity. Hence, we introduce a novel spanner property: the spanner *complexity*, that is, the total complexity of all edges in the spanner. Let S be a set of n point sites in a simple polygon P with m vertices. We present an algorithm to construct, for any constant $\varepsilon > 0$ and fixed integer $k \geq 1$, a $(2k + \varepsilon)$-spanner with complexity $O(mn^{1/k} + n \log^2 n)$ in $O(n \log^2 n + m \log n + K)$ time, where K denotes the output complexity. When we consider sites in a polygonal domain P with holes, we can construct such a $(2k + \varepsilon)$-spanner of similar complexity in $O(n^2 \log m + nm \log m + K)$ time. Additionally, for any constant $\varepsilon \in (0, 1)$ and integer constant $t \geq 2$, we show a lower bound for the complexity of any $(t - \varepsilon)$-spanner of $\Omega(mn^{1/(t-1)} + n)$.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases spanner, simple polygon, polygonal domain, geodesic distance, complexity

Digital Object Identifier 10.4230/LIPIcs.SoCG.2023.16

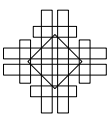
Related Version *Full Version*: https://arxiv.org/abs/2303.02997

1 Introduction

In the design of networks on a set of nodes, we often consider two criteria: few connections between the nodes, and small distances. Spanners are geometric networks on point sites that replace the small distance criterion by a small detour criterion. Formally, a *geometric t-spanner* for a set S of n point sites is an edge-weighted graph $\mathcal{G} = (S, E)$ for which the (weighted) distance $d_{\mathcal{G}}(p, q)$ between any two sites $p, q \in S$ is at most $t \cdot d(p, q)$, where $d(p, q)$ denotes the distance between p and q in the distance metric we consider [29]. The smallest value t for which a graph $\mathcal{G}$ is a t-spanner is called the *spanning ratio* of $\mathcal{G}$. The number of edges in the spanner is called the *size* of the spanner.

In the real world, spanners are often constructed in some sort of environment. For example, we might want to connect cities by a railway network, where the tracks should avoid obstacles such as mountains or lakes. One way to model such an environment is by a polygonal domain. In this paper, we study the case where the sites lie in a polygonal domain P with m vertices and h holes, and we measure the distance between two points p, q by their *geodesic distance*: the length of the shortest path between p and q fully contained within P. An example of such a spanner is provided in Figure 1.

The spanning ratio and the size of spanners are not the only properties of interest. Many different properties have been studied, such as total weight (or lightness), maximum degree, (hop) diameter, and fault-tolerance [4, 9, 11, 14, 20, 27, 28, 30]. When we consider distance metrics for which the edges in the spanner no longer have constant complexity, another



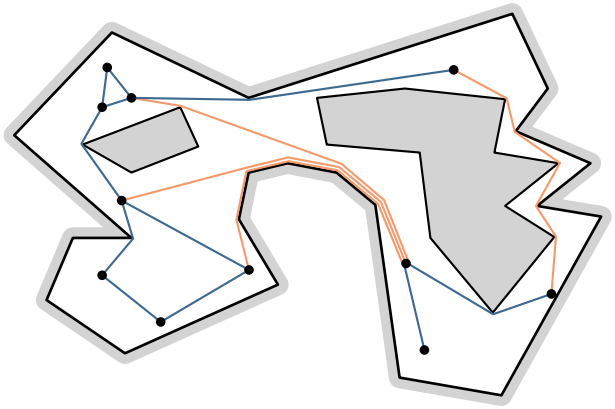


Figure 1 A spanner on a set of point sites in a polygonal domain. Because of the orange edges, the spanner has a relatively high complexity.

interesting property of spanners arises: the spanner *complexity*, i.e. the total complexity of all edges in the spanner. In our railway example, this corresponds to the total number of bends in the tracks. A spanner with a low number of bends may be desired, as trains can drive faster on straight tracks, and it makes construction cheaper. In this paper, we study this novel property for point sites in a polygonal domain, where the complexity of an edge is simply the number of line segments in the path. In this setting, a single edge may have complexity $\Theta(m)$. Naively, a spanner of size E could thus have complexity $\Theta(mE)$. Our goal is to compute an $O(1)$-spanner of size $O(n \operatorname{polylog} n)$ with small complexity, preferably near linear in both n and m.

When studying spanning trees of points, two variants exist: with or without Steiner points. The same is true for spanners, where Steiner points can be used to obtain lighter and sparser spanners [6, 27]. In this paper we focus on the variant where Steiner points are *not* allowed, leaving the other variant to future research.

Related work. For the Euclidean distance in $\mathbb{R}^d$, and any fixed $\varepsilon > 0$, there is a $(1+\varepsilon)$-spanner of size $O(n/\varepsilon^{d-1})$ [30]. For the more general case of metric spaces of bounded doubling dimension we can also construct a $(1+\varepsilon)$-spanner of size $O(n/\varepsilon^{O(d)})$ [13, 21, 24]. These results do not apply when the sites lie in a polygon, and we measure their distances using the geodesic distance. Abam et al. [1] show there is a set of n sites in a simple polygon P for which any geodesic $(2-\varepsilon)$-spanner has $\Omega(n^2)$ edges. They also construct a geodesic $(\sqrt{10}+\varepsilon)$-spanner of size $O(n \log^2 n)$ for sites in a simple polygon, and a geodesic $(5+\varepsilon)$-spanner of size $O(n\sqrt{h} \log^2 n)$ for sites in a polygonal domain. Recently, Abam et al. [3] showed that a geodesic $(2+\varepsilon)$-spanner with $O(n \log n)$ edges exists for points on a polyhedral terrain, thereby almost closing the gap between the upper and lower bound on the spanning ratio. However, they show only the existence of such a spanner, and leave constructing one open. Moreover, all of these spanners can have high, $\Omega(nm)$, complexity.

Abam et al. [3] make use of spanners on an *additively weighted* point set in $\mathbb{R}^d$. In this setting, the distance between two sites p, q is $w(p) + |pq| + w(q)$ for $p \neq q$, where $w(p)$ is the non-negative weight of a site $p \in S$ and $|pq|$ denotes the Euclidean distance, and 0 for $p = q$. Such additively weighted spanners were studied before by Abam et al. [2], who obtain an $O(5+\varepsilon)$-spanner of linear size, and an $O(2+\varepsilon)$-spanner of size $O(n \log n)$. They also provide a lower bound of $\Omega(n^2)$ on the size of any $(2-\varepsilon)$-spanner. Abam et al. [3] improve these results and obtain a nearly optimal additively weighted $(2+\varepsilon)$-spanner of size $O(n)$.

The other key ingredient for the geodesic $(2+\varepsilon)$-spanner of Abam et al. [3] is a *balanced shortest-path separator*. Such a separator consists of either a single shortest path between two points on the boundary of the terrain, or three shortest paths that form a *shortest-path triangle*. This separator partitions the terrain into two subterrains, and we call it balanced when each of these terrains contains roughly half of the sites in S. In their constructive proof for the existence of such a balanced separator, they assume that the three shortest paths in a shortest-path triangle are disjoint, except for their mutual endpoints. However, during their construction it can actually happen that these paths are *not* disjoint. When this happens, it is unclear exactly how to proceed. Just like for the $(2+\varepsilon)$-spanner, the computation of a balanced separator is left for future research. We show how to get rid of the assumption that the shortest paths are disjoint, and thereby confirm the result claimed by Abam et al. [3].

Next to spanners on the complete Euclidean geometric graph, spanners under line segment constraints were studied [8, 10, 12, 16, 17]. Here, the goal is to construct a spanner on the *visibility graph* of S with respect to a set of line segments between sites in S. If the segments form a polygonal domain P, this setting is similar to ours, except that *all* vertices of P are included as sites in S, and thus the complexity of each edge is constant.

Our results. We first consider the simple setting where the sites lie in a simple polygon, i.e. a polygonal domain without holes. We show that in this setting any $(3-\varepsilon)$-spanner may have complexity $\Omega(nm)$, thus implying that the $(2+\varepsilon)$-spanner of Abam et al. [3] may also have complexity $\Omega(nm)$, despite having $O(n \log n)$ edges.

To improve this complexity, we first introduce a simple 2-spanner with $O(n \log n)$ edges for an additively weighted point set in a 1-dimensional Euclidean space; see Section 2. In Section 3, we use this result to obtain a geodesic $2\sqrt{2}$-spanner with $O(n \log^2 n)$ edges for a point set in a simple polygon. We recursively split the polygon by a chord λ such that each subpolygon contains roughly half of the sites, and build a 1-dimensional spanner on the sites projected to λ. We then extend this spanner into one that also has bounded complexity. For any constant $\varepsilon > 0$ and fixed integer $k \geq 1$, we obtain a $(2k+\varepsilon)$-spanner with complexity $O(mn^{1/k} + n \log^2 n)$. Furthermore, we provide an algorithm to compute such a spanner that runs in $O(n \log^2 n + m \log n + K)$ time, where K denotes the output complexity. When we output each edge explicitly, K is equal to the spanner complexity. However, as each edge is a shortest path, we can also output an edge implicitly by only stating the two sites it connects. In this case K is equal to the size of the spanner. Additionally, for any constant $\varepsilon \in (0,1)$ and integer constant $t \geq 2$, we show a lower bound for the complexity of any $(t-\varepsilon)$-spanner of $\Omega(mn^{1/(t-1)} + n)$. Therefore, the $2k+\varepsilon$ spanning ratio of our $O(mn^{1/k} + n \log^2 n)$ complexity spanners is about a factor two off optimal.

In Section 4, we extend our results for a simple polygon to a polygonal domain. There are two significant difficulties in this transition: *(i)* we can no longer partition the polygon by a line segment such that each subpolygon contains roughly half of the sites, and *(ii)* the shortest path between two sites p, q may not be homotopic to the path from p to q via another site c. We solve problem *(i)* by using a shortest-path separator similar to Abam et al. [3]. To apply the shortest-path separator in a polygonal domain, we need new additional ideas. In particular, we allow one additional type of separator in our version of a shortest-path separator: two shortest paths from a point in P to the boundary of a single hole. We show that this way there indeed always exists such a separator in a polygonal domain, and provide an $O(n^2 \log m + nm \log m)$ time algorithm to compute one. To overcome problem *(ii)*, we allow an edge (p, q) to be any path from p to q. In networks, the connections between two nodes are often not necessarily optimal paths, the only requirement being that the distance between

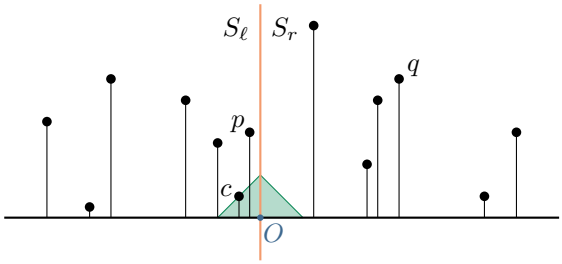


Figure 2 Construction of the additively weighted 1-dimensional spanner. The green triangle represents all points that are at distance at most $d_w(c, O)$ from O.

two hubs does not become too large. Thus allowing other paths between two sites seems a reasonable relaxation. This way, we obtain in a geodesic $(2k + \varepsilon)$-spanner of size $O(n \log^2 n)$ and complexity $O(mn^{1/k} + n \log^2 n)$ that can be computed in $O(n^2 \log m + nm \log m + K)$ time. Because our edges always consist of at most three shortest paths, we can again output the edges implicitly in $O(n \log^2 n)$ time. We also provide an alternative $(2k + \varepsilon)$-spanner of size $O(\sqrt{h}n \log^2 n)$ and complexity $O(\sqrt{h}(mn^{1/k} + n \log^2 n))$ that can be constructed more efficiently, i.e., in $O(\sqrt{h}n \log^2 n + m \log m + K)$ time.

Throughout the paper, we make the general position assumption that all vertices of P and sites in S have distinct x- and y-coordinates. Symbolic perturbation, in particular a shear transformation, can be used to remove this assumption [18].

2 A 1-dimensional additively weighted 2-spanner

We consider how to compute an additively weighted spanner $\mathcal{G}$ in 1-dimensional Euclidean space, where each site $p \in S$ has a non-negative weight $w(p)$. The distance $d_w(p, q)$ between two sites $p, q \in S$ is given by $d_w(p, q) = w(p) + |pq| + w(q)$, where $|pq|$ denotes the Euclidean distance. Without loss of generality, we can map $\mathbb{R}^1$ to the x-axis, and the weights to the y-axis, see Figure 2. This allows us to speak of the sites left (or right) of some site p.

To construct a spanner $\mathcal{G}$, we first partition the sites into two sets S_ℓ and S_r of roughly equal size by a point O with $w(O) = 0$. The set S_ℓ contains all sites left of O, and S_r all sites right of O. Sites that lie on the vertical line through O are not included in either of the sets. We then find a site $c \in S$ for which $d_w(c, O)$ is minimal. For all $p \in S$, $p \neq c$, we add the edge (p, c) to $\mathcal{G}$. Finally, we handle the sets S_ℓ and S_r, excluding the site c, recursively.

▶ **Lemma 1.** *The graph $\mathcal{G}$ is a 2-spanner of size $O(n \log n)$ and can be constructed in $O(n \log n)$ time.*

Proof sketch. For two sites p, q, consider the chosen center c at the level where p and q lie on different sides of O. As the shortest path from p to q passes via O, we have

$$d_{\mathcal{G}}(p, q) \leq d_w(p, O) + 2d_w(c, O) + d_w(q, O) \leq 2d_w(p, O) + 2d_w(q, O) = 2d_w(p, q). \quad \blacktriangleleft$$

3 Spanners in a simple polygon

3.1 A simple geodesic spanner

Just like Abam et al. [3], we use our 1-dimensional spanner to construct a geodesic spanner. We are more interested in the simplicity of the spanner than its spanning ratio, as we base our low complexity spanners, to be discussed in Section 3.2, on this simple geodesic spanner. Let

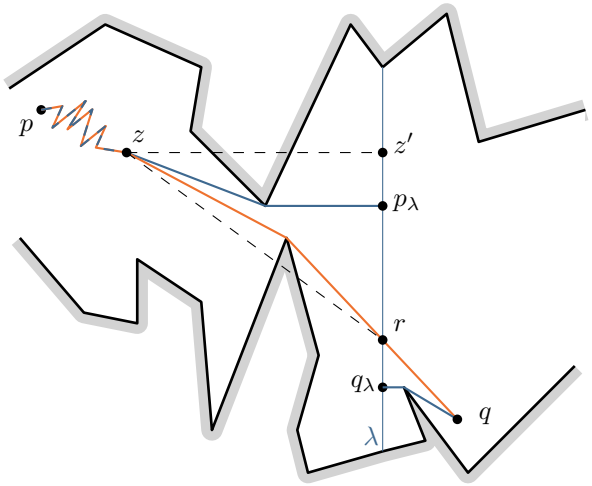


Figure 3 The shortest path $\pi(p, q)$ crosses λ at r. The difference in length between the direct path from z to r and the path through p_λ can be bounded by considering the triangle $\mathcal{T} = (z, z', r)$.

P be a simple polygon, and let ∂P denote the polygon boundary. We denote by $d(p, q)$ the geodesic distance between $p, q \in P$, and by $\pi(p, q)$ the shortest (geodesic) path from p to q. We analyze the simple construction using any 1-dimensional additively weighted t-spanner of size $O(n \log n)$. We show that restricting the domain to a simple polygon improves the spanning ratio from $3t$ to $\sqrt{2}t$. The construction can be refined to achieve a spanning ratio of $t + \varepsilon$, see Section 3.2.2 and Lemma 3 of the full version [19].

As in [1] and [3], we first partition P into two subpolygons P_ℓ and P_r by a line segment λ, such that each subpolygon contains at most two thirds of the sites in S [7]. We assume, without loss of generality, that λ is a vertical line segment and P_ℓ is left of λ. Let S_ℓ be the sites in the closed region P_ℓ, and $S_r := S \setminus S_\ell$. For each site $p \in S$, we then find the point p_λ on λ closest to p. Note that this point is unique, because the shortest path to a line segment is unique in a simple polygon. We denote by S_λ the set of all projected sites. As λ is a line segment, we can define a weighted 1-dimensional Euclidean space on λ, where $w(p_\lambda) := d(p, p_\lambda)$ for each $p_\lambda \in S_\lambda$. We compute a t-spanner $\mathcal{G}_\lambda = (S_\lambda, E_\lambda)$ for this set. For each pair $(p_\lambda, q_\lambda) \in E_\lambda$, we add the edge (p, q), which is $\pi(p, q)$, to our spanner $\mathcal{G}$. Finally, we recursively compute spanners for S_ℓ and S_r, and add their edges to $\mathcal{G}$ as well.

▶ **Lemma 2.** *The graph $\mathcal{G}$ is a geodesic $\sqrt{2}t$-spanner of size $O(n \log^2 n)$.*

Proof. As $\mathcal{G}_\lambda$ has $O(n \log n)$ edges (Lemma 1) that directly correspond to edges in $\mathcal{G}$, and the recursion has $O(\log n)$ levels, we have $O(n \log^2 n)$ edges in total. Let p, q be two sites in S. If both are in S_ℓ (or S_r), then there is a path of length $\sqrt{2}td(p, q)$ by induction. So, we assume w.l.o.g. that $p \in S_\ell$ and $q \in S_r$. Let r be the intersection point of $\pi(p, q)$ and λ. Observe that p_λ and q_λ must be on opposite sides of r, otherwise r cannot be on the shortest path. We assume, without loss of generality, that p_λ is above r and q_λ below r. Because $\mathcal{G}_\lambda$ is a t-spanner, we know that there is a weighted path from p_λ to q_λ of length at most $td_w(p_\lambda, q_\lambda)$. As $w(p_\lambda) = d(p, p_\lambda)$, this directly corresponds to a path in the polygon. So,

$$d_{\mathcal{G}}(p, q) \leq d_{\mathcal{G}_\lambda}(p_\lambda, q_\lambda) \leq td_w(p_\lambda, q_\lambda) = t(d(p, p_\lambda) + |p_\lambda r| + |rq_\lambda| + d(q_\lambda, q)). \tag{1}$$

Let z be the point where the shortest paths from p to p_λ and r separate. See Figure 3 for an illustration. Consider the right triangle $\mathcal{T} = (z, z', r)$, where z' is the intersection point

of the line perpendicular to λ through z and the line containing λ. Note that z' does not necessarily lie within P. For this triangle we have that

$$|zr| \geq \frac{\sqrt{2}}{2}(|zz'| + |z'r|). \tag{2}$$

Next, we show that the path from z to p_λ is a y-monotone convex polygonal chain ending at or below z'. Consider the vertical ray through z upwards to the polygon boundary. We call the part of ∂P between where the ray hits ∂P and λ the *top* part of ∂P. Similarly, for a downwards ray, we define the *bottom* part of ∂P. There are no vertices on $\pi(z, p_\lambda)$ from the bottom part of ∂P, because such a vertex would then also occur on the shortest path to r. This is in contradiction with the definition of z. If z sees z', then $p_\lambda = z'$, otherwise the chain must bend at one or more vertices of the top part of ∂P, and thus lie below z'. It follows that $\pi(z, p_\lambda)$ is contained within $\mathcal{T}$. Similarly, we conclude that $\pi(z, r)$ is contained within $\mathcal{T}$. Additionally, this gives us that $d(z, p_\lambda) \leq |zz'| + |z'p_\lambda|$, and $d(z, r) \geq |zr|$. Together with Equation (2) this yields $d(z, p_\lambda) + |p_\lambda r| \leq |zz'| + |z'r| \leq \sqrt{2}|zr| \leq \sqrt{2}d(z, r)$. And thus

$$d(p, p_\lambda) + |p_\lambda r| = d(p, z) + d(z, p_\lambda) + |p_\lambda r| \leq d(p, z) + \sqrt{2}d(z, r) \leq \sqrt{2}d(p, r).$$

Symmetrically, we find for q that $d(q, q_\lambda) + |q_\lambda r| \leq \sqrt{2}d(q, r)$. From this, together with Equation (1), we conclude that $d_{\mathcal{G}}(p, q) \leq t\left(\sqrt{2}d(p, r) + \sqrt{2}d(r, q)\right) = \sqrt{2}td(p, q)$. ◀

Applying Lemma 2 to the spanner of Section 2 yields a $2\sqrt{2}$-spanner of size $O(n \log^2 n)$.

3.2 Low complexity geodesic spanners

In general, a geodesic spanner $\mathcal{G} = (S, E)$ in a simple polygon with m vertices may have complexity $O(m|E|)$. It is easy to see that the $2\sqrt{2}$-spanner of Section 3.1 can have complexity $\Omega(nm)$, just like the spanners in [3]. As one of the sites, c, is connected to all other sites, the polygon in Figure 4 provides this lower bound. The construction in Figure 4 even shows that the same lower bound holds for the complexity of any $(3 - \epsilon)$-spanner. Additionally, the following theorem implies a trade-off between the spanning ratio and the spanner complexity.

▶ **Theorem 3.** *For any constant $\varepsilon \in (0, 1)$ and integer constant $t \geq 2$, there exists a set of n point sites in a simple polygon P with $m = \Omega(n)$ vertices for which any geodesic $(t - \varepsilon)$-spanner has complexity $\Omega(mn^{1/(t-1)})$.*

The proofs of these lower bounds are in the full version [19]. Next, we present a spanner that almost matches this bound. We first present a $4\sqrt{2}$-spanner of bounded complexity, and then generalize the approach to obtain a $(2k + \varepsilon)$-spanner of complexity $O(mn^{1/k} + n \log^2 n)$, for any integer $k \geq 2$.

3.2.1 A $4\sqrt{2}$-spanner of complexity $O(m\sqrt{n} + n \log^2 n)$

To improve the complexity of the geodesic spanner, we adapt our construction for the additively weighted spanner $\mathcal{G}_\lambda$ as follows. After finding the site $c_\lambda \in S_\lambda$ for which $d_w(c_\lambda, O)$ is minimal, we do not add all edges (p_λ, c_λ), $p_\lambda \in S_\lambda$, to $\mathcal{G}_\lambda$. Instead, we form groups of sites whose original sites (before projection) are "close" to each other in P. For each group S_i, we add all edges $(p_\lambda, c_{i,\lambda})$, $p_\lambda \in S_i$, to $\mathcal{G}_\lambda$, where $c_{i,\lambda}$ is the site in S_i for which $d_w(c_{i,\lambda}, O)$ is minimal. Finally, we add all edges $(c_{i,\lambda}, c_\lambda)$ to $\mathcal{G}_\lambda$.

To make sure the complexity of our spanner does not become too large, we must choose the groups in such a way that the spanner edges do not cross "bad" parts of P too often. The following lemma states the properties that we require of our groups to achieve this.

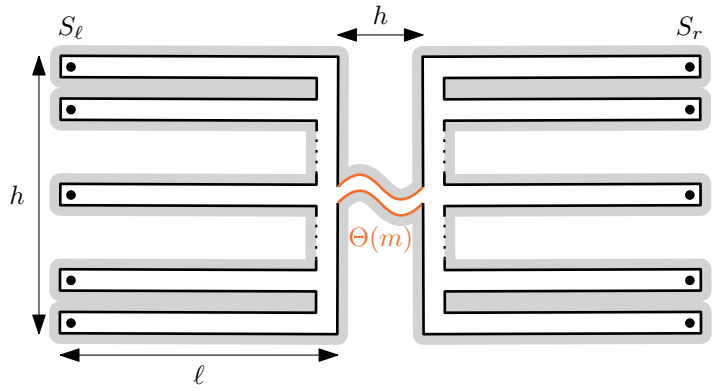


Figure 4 Any $(3-\varepsilon)$-spanner in a simple polygon with m vertices may have complexity $\Omega(nm)$.

▶ **Lemma 4.** *If the groups adhere to the following properties, then $\mathcal{G}$ has $O(m\sqrt{n}+n\log^2 n)$ complexity:*

1. *each group contains $\Theta(\sqrt{n})$ sites, and*
2. *each vertex of P is only used by shortest paths within $O(1)$ groups.*

Proof. We will first prove the complexity of the edges in one level of the 1-dimensional spanner is $O(m\sqrt{n}+n)$. Two types of edges are added to the spanner: (a) edges from some c_i to c, and (b) edges from some $p \in S_i$ to c_i. According to property 1, there are $\Theta(\sqrt{n})$ groups, and thus $\Theta(\sqrt{n})$ type (a) edges, that each have a complexity of $O(m)$. Thus the total complexity of these edges is $O(m\sqrt{n})$. Let r_i be the maximum complexity of a shortest path between any two sites in S_i and let V_i be the set of vertices this path visits. Property 2 states that for any $v \in V_i$ it holds that $|\{j \mid v \in V_j\}| = O(1)$, which implies that $\sum_i r_i = O(m)$. The complexity of all type (b) edges is thus $O(n) + \sum_i r_i O(\sqrt{n}) = O(m\sqrt{n}+n)$.

Next, we show that in both recursions, the 1-dimensional recursion and the recursion on P_ℓ and P_r, not only the number of sites, but also the complexity of the polygon is divided over the two subproblems. Splitting the sites into left and right of O corresponds to splitting the polygon horizontally at O: all sites left (right) of O in the 1-dimensional space lie in the part of the polygon below (above) this horizontal line segment. Thus, shortest paths between sites left of O use part of the polygon that is disjoint from the shortest paths between the sites right of O. This means that for two subproblems we have that $m_1 + m_2 = m$, where m_i denotes the maximum complexity of a path in subproblem i. The recursion for the complexity is now given by

$$T(n,m) = T(n/2, m_1) + T(n/2, m_2) + O(m\sqrt{n}+n), \text{ with } m_1 + m_2 = m.$$

In the full version [19] we show this solves to $T(n) = O(m\sqrt{n} + n\log n)$.

Similarly, the split by λ divides the polygon into two subpolygons, while adding at most two new vertices. As all vertices, except for the endpoints of λ, are in P_ℓ or P_r (not both), the total complexity of both subpolygons is at most $m+4$. We obtain the following recursion

$$T(n,m) = T(n/2, m_1) + T(n/2, m_2) + O(m\sqrt{n} + n\log n), \text{ with } m_1 + m_2 = m + 4,$$

which solves to $T(n) = O(m\sqrt{n} + n\log^2 n)$. ◀

To form groups that adhere to these two properties, we consider the shortest path tree SPT_c of c: the union of all shortest paths from c to the vertices of P. We include the sites $p \in S \setminus \{c\}$ as leaves in SPT_c as children of their apex, i.e., the last vertex on $\pi(c,p)$. This

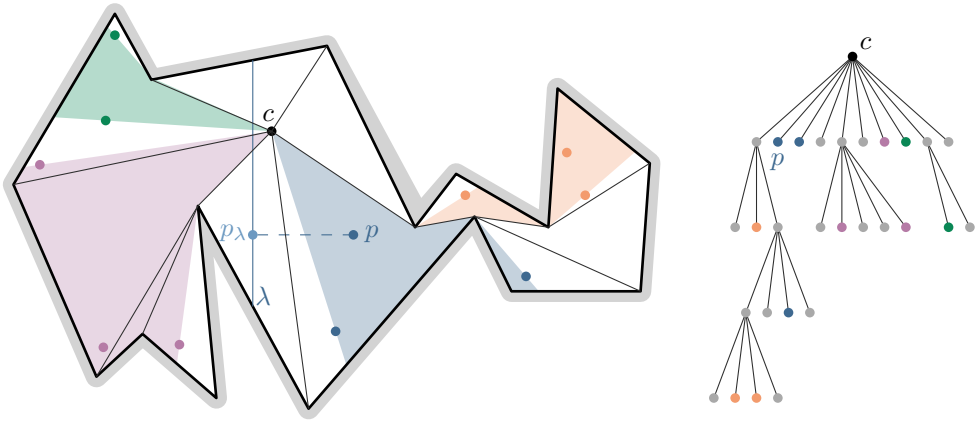


Figure 5 The shortest path tree of c. Each group S_i has an associated polygonal region R_i in P.

gives rise to an ordering of the sites in S, and thus of the weighted sites in S_λ, based on the in-order traversal of the tree. We assign the first $\lceil\sqrt{n}\rceil$ sites to S_1, the second $\lceil\sqrt{n}\rceil$ to S_2, etc. See Figure 5.

Clearly these groups adhere to property 1. Proving that they also adhere to property 2 is more involved. For each group S_i, consider the minimal subtree $\mathcal{T}_i$ of SPT_c containing all $p \in S_i$. $\mathcal{T}_i$ defines a polygonal region R_i in P as follows. Refer to Figure 5 for an illustration. Let v_i be the root of $\mathcal{T}_i$. Consider the shortest path $\pi(v_i, a)$, where a is the first site of S_i in $\mathcal{T}_i$ by the ordering used before. Let π_a be the path obtained from $\pi(v_i, a)$ by extending the last segment of $\pi(v_i, a)$ to the boundary of P. Similarly, let π_b be such a path for the last site of S_i in $\mathcal{T}_i$. We take R_i to be the region in P rooted at v_i and bounded by π_a, π_b, and some part of the boundary of P, that contains the sites in S_i. In case v_i is c, we split R_i into two regions R_j and R_k, such that the angle of each of these regions at c is at most π. The set S_i is then also split into two sets S_j and S_k accordingly. The following three lemmas on R_i and $\mathcal{T}_i$ together imply that the groups adhere to property 2.

▶ **Lemma 5.** *Only vertices of P that are in $\mathcal{T}_i$ can occur in R_i.*

▶ **Lemma 6.** *All shortest paths between sites in S_i are contained within R_i.*

▶ **Lemma 7.** *Any vertex $v \in SPT_c$ occurs in at most two trees $\mathcal{T}_i$ and $\mathcal{T}_j$ as a non-root node.*

Note that the root r of $\mathcal{T}_i$ is never used in a shortest path between sites in S_i, because r cannot be a reflex vertex of R_i. Consequently, Lemma 4 states that the spanner has complexity $O(m\sqrt{n} + n\log^2 n)$.

▶ **Lemma 8.** *The graph $\mathcal{G}$ is a geodesic $4\sqrt{2}$-spanner of size $O(n\log^2 n)$.*

Proof. We prove the 1-dimensional spanner $\mathcal{G}_\lambda$ is a 4-spanner with $O(n\log n)$ edges. Together with Lemma 2, this directly implies $\mathcal{G}$ is a $4\sqrt{2}$-spanner with $O(n\log^2 n)$ edges.

In each level of the recursion, we still add only a single edge for each site. Thus, the total number of edges is $O(n\log n)$. Again, consider two sites $p_\lambda, q_\lambda \in S_\lambda$, and let c_λ be the chosen center point at the level where p_λ and q_λ are separated by O. Let S_i be the group of p_λ and S_j the group of q_λ. Both the edges $(p_\lambda, c_{i,\lambda})$ and $(c_{i,\lambda}, c_\lambda)$ are in $\mathcal{G}_\lambda$, similarly for q_λ. We thus have a path $p_\lambda \to c_{i,\lambda} \to c_\lambda \to c_{j,\lambda} \to q_\lambda$ in $\mathcal{G}_\lambda$. Using that $d_w(p_\lambda, c_{i,\lambda}) \leq d_w(p_\lambda, O) + d_w(c_{i,\lambda}, O)$, because of the triangle inequality, and $d_w(c_{i,\lambda}, O) \leq d_w(p_\lambda, O)$, we find:

$$\begin{aligned} d_{\mathcal{G}_\lambda}(p_\lambda, q_\lambda) &= d_w(p_\lambda, c_{i,\lambda}) + d_w(c_{i,\lambda}, c_\lambda) + d_w(c_\lambda, c_{j,\lambda}) + d_w(c_{j,\lambda}, q_\lambda) \\ &\leq d_w(p_\lambda, O) + 2d_w(c_{i,\lambda}, O) + 2d_w(c_\lambda, O) + 2d_w(c_{j,\lambda}, O) + d_w(q_\lambda, O) \\ &\leq 4d_w(p_\lambda, O) + 4d_w(q_\lambda, O) \\ &= 4d_w(p_\lambda, q_\lambda) \end{aligned}$$ ◀

3.2.2 A $(2k+\varepsilon)$-spanner of complexity $O(mn^{1/k} + n\log^2 n)$

In this section we sketch how to generalize the approach of Section 3.2.1 to obtain a spanner with a trade-off between the (constant) spanning ratio and complexity. Fix $N = n^{1/k}$, for some integer constant $k \geq 1$. Instead of $\Theta(\sqrt{n})$ groups, we create $\Theta(N)$ groups. For each of these groups we select a center, and then partition the groups further recursively. By connecting each center to its parent center, we obtain a tree of height k. This results in a spanning ratio of $k2\sqrt{2}$.

Abam et al. [3] refine their spanner construction to obtain a $(2+\varepsilon)$-spanner. We generalize this refinement in Lemma 3 of the full version [19]. The main idea is as follows. For each point p_λ, we additionally include a collection of $O(t^2/\varepsilon^2)$ evenly spread points on λ close to p_λ in S_λ. For an edge in $\mathcal{G}_\lambda$ between a point in the collection of p_λ and a point in the collection of q_λ, we add the edge (p, q) to $\mathcal{G}$. This way, we even obtain a $(2k+\varepsilon)$-spanner.

▶ **Lemma 9.** *For any constant $\varepsilon > 0$ and integer constant $k \geq 1$, there exists a geodesic $(2k+\varepsilon)$-spanner of size $O(c_{\varepsilon,k} n \log^2 n)$ and complexity $O(c_{\varepsilon,k}(mn^{1/k} + n\log^2 n))$, where $c_{\varepsilon,k}$ is a constant depending only on ε and k.*

3.3 Construction algorithm

In this section we propose an algorithm to construct the spanners of Section 3.2. The following gives a general overview of the algorithm, which computes a $(2k+\varepsilon)$-spanner in $O(n\log^2 n + m\log n)$ time. In the rest of this section we will discuss the steps in more detail.

1. Preprocess P for efficient shortest path queries and build both the vertical decomposition $\mathcal{VD}$ and horizontal decomposition $\mathcal{HD}$ of P.
2. For each $p \in S$, find the trapezoid in $\mathcal{VD}$ and $\mathcal{HD}$ that contains p. For each trapezoid $\nabla \in \mathcal{VD}$, store the number of sites of S that lies in ∇ and sort these sites on their x-coordinate.
3. Recursively compute a spanner on the sites S in P:
   a. Find a vertical chord λ of P such that λ partitions P into two polygons P_ℓ and P_r, and each subpolygon contains at most $2n/3$ sites using the algorithm of Lemma 10.
   b. For each $p \in S$, find the point p_λ on λ and its weight using the algorithm of Lemma 11, and add this point to S_λ.
   c. Compute an additively weighted 1-dimensional spanner $\mathcal{G}_\lambda$ on the set S_λ using the algorithm of Lemma 12 or Lemma 13.
   d. For every edge $(p_\lambda, q_\lambda) \in E_\lambda$ add the edge (p, q) to $\mathcal{G}$.
   e. Recursively compute spanners for S_ℓ in P_ℓ and S_r in P_r.

In step 1, we preprocess the polygon in $O(m)$ time such that the distance between any two points $p, q \in P$ can be computed in $O(\log m)$ time [15, 22]. We also build the horizontal and vertical decompositions of P, and a corresponding point location data structure, as a preprocessing step in $O(m)$ time [15, 26]. We then perform a point location query for each site $p \in S$ in $O(n\log m)$ time in step 2 and sort the sites within each trapezoid in $O(n\log n)$

time in total. The following lemma describes the algorithm to compute a vertical chord that partitions P into two subpolygons such that each of them contains roughly half of the sites in S. It is based on the algorithm of Bose et al. [7] that finds such a chord without the constraint that it should be vertical. Because of this constraint, we use the vertical decomposition of P instead of a triangulation in our algorithm.

▶ **Lemma 10.** *In $O(n+m)$ time, we can find a vertical chord of P that partitions P into two subpolygons P_ℓ and P_r, such that each subpolygon contains at most $2n/3$ sites of S.*

The following lemma states that we can find the projections p_λ efficiently. The algorithm produces not only these projected sites, but also the shortest path tree SPT_λ of λ.

▶ **Lemma 11.** *We can compute the closest point p_λ on λ and $d(p, p_\lambda)$ for all sites $p \in S$, and the shortest path tree SPT_λ, in $O(m + n \log m)$ time.*

▶ **Lemma 12.** *Given SPT_λ, we can construct a 4-spanner $\mathcal{G}_\lambda$ on the additively weighted points S_λ, where the groups adhere to the properties of Lemma 4, in $O(n \log n + m)$ time.*

Proof. The 4-spanner of Section 3.2.1 requires an additional step at each level of the recursion, namely the formation of $\Theta(\sqrt{n})$ groups. We first discuss the running time to construct a 4-spanner when forming the groups as in Section 3.2.1, and then improve the running time by introducing a more efficient way to form the groups.

In Section 3.2.1, the groups are formed based on the shortest path tree of the site c. Building the shortest path tree, and a corresponding point location data structure, takes $O(m)$ time [23]. Then, we perform a point location query for each site to find its apex in the shortest path tree, and add the sites to the tree. These queries take $O(n \log m)$ time in total. We form groups based on the traversal of the tree. Note that we do not distinguish between sites with the same parent in the tree, as the tree $\mathcal{T}_i$ (and thus the region R_i) obtained contains the same vertices of P regardless of the order of these sites. After obtaining the groups, we again add only $O(n)$ edges to the spanner. The overall running time of the algorithm is thus $O((m + n \log m) \log n)$.

This running time can be improved by using another approach to form the groups. To form groups that adhere to the properties of Lemma 4, and thus result in a spanner of the same complexity, we can use any partition of P into regions R_i, as long R_i as contains $\Theta(\sqrt{n})$ sites and $\sum_i r_i = O(m)$. Next, we describe how to form such groups efficiently using SPT_λ.

We first define an ordering on the sites. This is again based on the traversal of some shortest path tree. Instead of considering the shortest path tree of a point site, we consider the shortest path tree SPT_λ of λ. Again, all sites in S are included in this shortest path tree. Additionally, we split the node corresponding to λ into a node for each distinct projection point on λ (of the vertices and the sites) and add an edge between each pair of adjacent points, see Figure 6. We root the tree at the node corresponding to the bottom endpoint of λ. Whenever a node t on λ has multiple children, in other words, when multiple sites are projected to the same point t, our assumption that all y-coordinates are distinct ensures that all these sites lie either in P_ℓ or P_r.

The groups are formed based on the in-order traversal of this tree, which can be performed in $O(m + n)$ time. As before, the first $\lceil \sqrt{n} \rceil$ are in S_1, the second in S_2, etc. The groups thus adhere to the first property. Next, we show they also adhere to the second property.

For each group S_i, we again consider the minimal subtree $\mathcal{T}_i$ of SPT_λ containing all $p \in S_i$. $\mathcal{T}_i$ defines a region R_i in P as follows. Let a be the first site of S_i in $\mathcal{T}_i$ by the ordering used before. Assume that a lies in P_ℓ. We distinguish two cases: $a_\lambda \in \mathcal{T}_i$, or $a_\lambda \notin \mathcal{T}_i$. When $a_\lambda \in \mathcal{T}_i$, then let π_a be the path obtained from $\pi(a_\lambda, a)$ by extending the last segment to

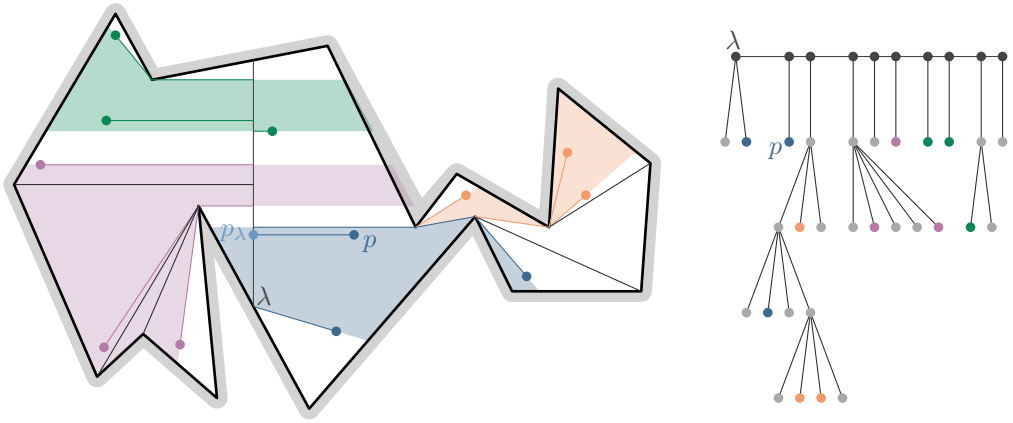


Figure 6 The shortest path tree SPT_λ and associated polygonal region R_i for each group S_i.

the boundary of P. Additionally, we extend π_a into P_r horizontally until we hit the polygon boundary. When $a_\lambda \notin \mathcal{T}_i$, consider the root v_i of $\mathcal{T}_i$. Let π_a be the path obtained from $\pi(v_i, a)$ by extending the last segment of the path to the boundary of P. Similarly, let π_b be such a path for the rightmost site of S_i in $\mathcal{T}_i$. We take R_i to be the region in P bounded by π_a, π_b, and some part of the boundary of P that contains the sites in S_i. See Figure 6. Note that, as before, only vertices of P that are in $\mathcal{T}_i$ can occur in R_i. All shortest paths between sites in S_i are contained within R_i. Just as for the shortest path tree of c, Lemma 7 implies that any vertex $v \in SPT_\lambda$ occurs in at most two trees $\mathcal{T}_i$ and $\mathcal{T}_j$ as a non-root vertex. We conclude that any vertex is used by shortest paths within at most two groups.

After splitting λ at a point O, the tree SPT_λ is also split into two trees $\mathcal{T}_\ell$ and $\mathcal{T}_r$ that contain exactly the sites in S_ℓ and S_r. We can thus reuse the ordering to form groups at each level of the recursion. This way, the total running time at a single level of the recursion is reduced to $O(n)$. The overall running time thus becomes $O(n \log n + m)$. ◀

▶ **Lemma 13.** *Given SPT_λ, we can construct a $2k$-spanner $\mathcal{G}_\lambda$ on the additively weighted points S_λ, where groups are formed as in Section 3.2.2, in $O(n \log n + m)$ time.*

Proof. To construct the $(2k + \varepsilon)$-spanner of Section 3.2.2, we can use the shortest path tree of λ to form the groups as before. Note that we can select a center for each group after computing the groups, as including the center in the subgroups does not influence spanning ratio or complexity. After ordering the sites based on the in-order traversal of SPT_λ, we can build the tree of groups in linear time using a bottom up approach. As before, fix $N = n^{1/k}$. We first form the $\Theta(N^k)$ lowest level groups, containing only a single site, and select a center for each group. Each group at level i is created by merging $\Theta(N)$ groups at level $i-1$, based on the same ordering. We do not perform this merging explicitly, but for each group we select the site closest to O of the merged level-$(i-1)$ centers as the center. Because our center property, being the closest to O, is decomposable, this indeed gives us the center of the entire group. This way, we can compute the edges added in one level of the recursion in linear time, so the running time remains $O(n \log n + m)$. ◀

The total running time thus becomes $O((n(\log n + \log m) + m) \log n) = O(n \log^2 n + m \log n)$. By splitting the polygon alternately based on the sites and the polygon vertices, we can replace the final $O(\log n)$ factor by $O(\min(\log n, \log m))$. Using the refinement of Section 3.2.2, step 3 is performed on $c_{\varepsilon,k} n$ sites, where $c_{\varepsilon,k}$ is a constant depending only on ε and k, increasing the running time for this step by a factor $O(c_{\varepsilon,k})$. Together with Lemma 9, we obtain the following theorem.

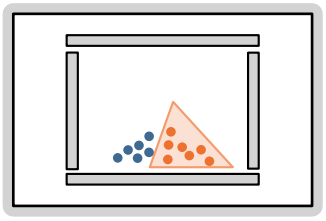


Figure 7 No shortest path between two points on P can separate the sites into two groups. We can separate the sites using three shortest paths, for example using the orange triangle.

▶ **Theorem 14.** *Let S be a set of n point sites in a simple polygon P with m vertices, and let $k \geq 1$ be any integer constant. For any constant $\varepsilon > 0$, we can build a geodesic $(2k+\varepsilon)$-spanner of size $O(c_{\varepsilon,k} n \log^2 n)$ and complexity $O(c_{\varepsilon,k}(mn^{1/k}+n\log^2 n))$ in $O(c_{\varepsilon,k} n \log^2 n + m \log n + K)$ time, where $c_{\varepsilon,k}$ is a constant depending only on ε and k, and K is the output complexity.*

4 Spanners in a polygonal domain

We consider a set of point sites S that lie in a polygonal domain P with m vertices and h holes. Let ∂P denote the boundary of the outer polygon. Suppose we would try the same construction as we used for a simple polygon in Section 3. Then, in a polygonal domain, we run into two problems. First, we cannot split the polygonal domain into two subpolygons by a line segment λ that roughly splits the sites in S, because any line segment that appropriately partitions S might intersect one or more holes. Second, the shortest path $\pi(p, q)$ between two sites $p, q \in S$ is no longer homotopic to a path $\pi(p, c) \cup \pi(c, q)$, which means our bound on the complexity is no longer valid. In the next section, we describe these problems and how we overcome them in more detail.

4.1 Low complexity geodesic spanners

Subdividing the domain. To apply our 1-dimensional spanner, we require only that the splitting curve λ is a shortest path in P. Instead of partitioning the domain by a line segment, we can thus use one or more shortest paths to partition the domain. Allowing only a shortest path between two points on ∂P as our separator is not sufficient, as it is not always possible to split the sites roughly equally this way. See Figure 7 for an example. Therefore, we use a *balanced shortest-path separator* (sp-separator), based on the balanced sp-separator of Abam et al. [3].

We use three types of separator: a shortest path between two points on ∂P (1-separator), two shortest paths starting at the same point and ending at the boundary of a single hole (2-separator), and three shortest paths $\pi(u, v)$, $\pi(v, w)$, and $\pi(u, w)$ with $u, v, w \in P$ (3-separator). Let P_ℓ be the polygonal domain to the left of λ, when λ is a 1-separator, and interior to λ, when λ is a 2- or 3-separator. Symmetrically, P_r is the domain to the right of, or exterior to, λ. See Figure 8 for an illustration. In the full version [19], we prove the existence of such a separator. In the proof, we correct for a missing case in the proof by Abam et al. [3] for the existence of a separator on a polyhedral terrain, thereby confirming their result. We also require an additional step in the constructive proof, because unlike a terrain, a polygonal domain is not continuous. This is where a 2-separator is required, which does not occur on a terrain. Finally, we provide an algorithm to construct a balanced sp-separator in a polygonal domain. This results in the following theorem.

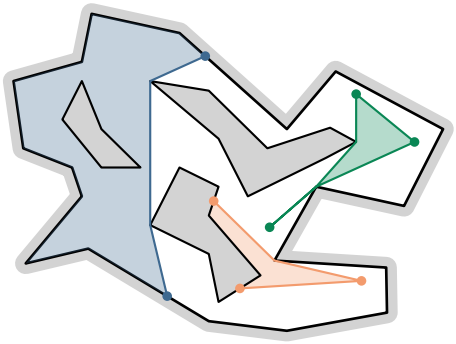


Figure 8 The three types of separators: A 1-separator (blue), a 2-separator (orange), and a 3-separator (green). For each separator P_ℓ is the colored region.

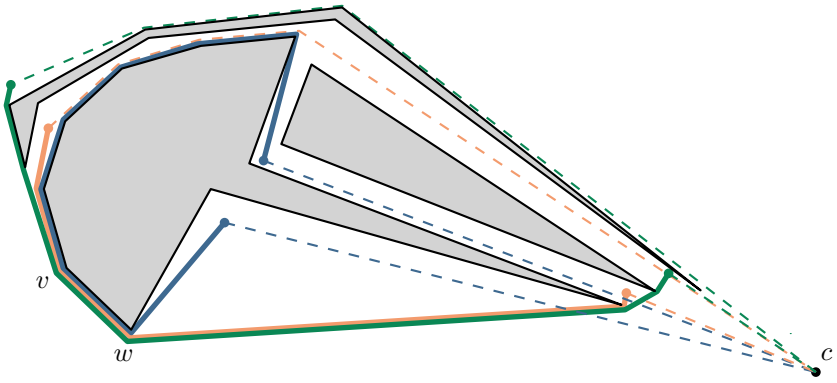


Figure 9 Assigning the sites to groups based on the shortest path tree of c, as described in Section 3.2.1, forms these colored groups. The shortest path from each site to c is shown dashed. Each shortest path between the two sites of a group visits both vertices v and w.

▶ **Theorem 15.** *Let S be a set of n point sites in a polygonal domain P with m vertices. A balanced sp-separator exists and it can be computed in $O(n^2 \log m + nm \log m)$ time.*

The spanner edges. In Section 3.2.1 and 3.3, we form groups of sites such that the shortest paths within a group are (almost) disjoint from the shortest paths within other groups. We use the shortest path tree of c (or λ) to obtain these groups. In a polygonal domain, the paths $\pi(p, q)$ and $\pi(p, c) \cup \pi(c, q)$, with $p, q, c \in P$, are not necessarily homotopic. Figure 9 gives an example of how this can result in many groups that use the same vertex of P in their shortest paths.

So far, we assumed that every edge $(p, q) \in E$ is a shortest path between p and q. To obtain a spanner of low complexity, we can also allow an edge between p and q to be any path between the two sites, as long as we keep the desired spanning ratio. Note that our complexity lower bounds still hold in this case. In particular, for an edge (p_λ, q_λ) in the 1-dimensional spanner, the edge (p, q) that we add to $\mathcal{G}$ is no longer $\pi(p, q)$. Instead, let (p, q) be the shortest path from p to q via p_λ and q_λ, excluding any overlap of the path. We denote this path by $\pi_\lambda(p, q)$. Formally, $\pi_\lambda(p, q)$ is defined as follows.

▶ **Definition 16.** *The path $\pi_\lambda(p, q)$ is given by:*

- $\pi(p, p_\lambda) \cup \pi(p_\lambda, q_\lambda) \cup \pi(q_\lambda, q)$, *where $\pi(p_\lambda, q_\lambda) \subseteq \lambda$, if $\pi(p, p_\lambda)$ and $\pi(q, q_\lambda)$ are disjoint,*
- $\pi(p, r) \cup \pi(r, q)$, *where r denotes the closest point to p of $\pi(p, p_\lambda) \cap \pi(q, q_\lambda)$, otherwise.*

In the full version [19], we show that using these paths as edges does not increase the spanning ratio, and prove a similar result to Lemma 4 for this new edge type. Additionally, we prove that using the shortest path tree of λ for the formation of the groups ensures that any vertex of P is used by paths $\pi_\lambda(p, q)$ within only $O(1)$ groups.

4.2 Construction algorithm

As before, let S_ℓ be the sites in the closed region P_ℓ, and $S_r := S \setminus S_\ell$. The following gives an overview of the algorithm that computes a $(2k+\varepsilon)$-spanner of complexity $O(mn^{1/k} + n\log^2 n)$ in $O(n^2 \log m + nm \log m)$ time for a set of point sites S in a polygonal domain P.

1. Find an sp-separator such that P is partitioned into two polygons P_ℓ and P_r, and S_ℓ contains at least $2n/9$ and at most $2n/3$ sites using the algorithm of Theorem 15.
2. For each shortest path λ of the separator:
   a. For each $p \in S$ find the weighted point p_λ on λ and add this point to S_λ.
   b. Compute an additively weighted 1-dimensional spanner $\mathcal{G}_\lambda$ on the set S_λ.
   c. For every edge $(p_\lambda, q_\lambda) \in E_\lambda$ add the edge $(p, q) = \pi_\lambda(p, q)$ to $\mathcal{G}$.
3. Recursively compute spanners for S_ℓ in P_ℓ and S_r in P_r.

According to Theorem 15, step 1 takes $O(n^2 \log m + nm \log m)$ time. In step 2, we find the projected sites using the shortest path map of λ in $O((m + n) \log m)$ time [25], and then compute the 1-dimensional spanner in $O(n \log n + m)$ time as in Lemma 13. This means that the construction of the sp-separator dominates the construction time of the spanner.

▶ **Theorem 17.** *Let S be a set of n point sites in a polygonal domain P with m vertices, and let $k \geq 1$ be any integer constant. For any constant $\varepsilon > 0$, we can build a geodesic $(2k + \varepsilon)$-spanner of size $O(c_{\varepsilon,k} n \log^2 n)$ and complexity $O(c_{\varepsilon,k}(mn^{1/k} + n \log^2 n))$ in $O(c_{\varepsilon,k}(n^2 \log m + nm \log m) + K)$ time, where $c_{\varepsilon,k}$ is a constant depending only on ε and k, and K is the output complexity.*

We can no longer output the edges implicitly by stating only the endpoints of the edges. Instead, we output an edge $\pi_\lambda(p, q)$ implicitly by stating the points p_λ and q_λ, when $\pi(p, p_\lambda)$ and $\pi(q, q_\lambda)$ are disjoint, or the point r from Definition 16 when they are not disjoint. The point r is the lowest common ancestor of the nodes p and q in SPT_λ. This can be computed in $O(1)$ time after preprocessing SPT_λ in $O((n + m) \log(n + m))$ [5].

4.3 A $(2 + \varepsilon)$-spanner with a dependence on $\sqrt{h}$

Because the computation of a balanced shortest-path separator is quite expensive, we consider another method to partition the domain by Abam et al. [1] using $\sqrt{h}$ simple polygons, where h is the number of holes in P. In the full version [19], we show this indeed improves the construction time, albeit at an increase in complexity.

▶ **Theorem 18.** *Let S be a set of n point sites in a polygonal domain P with m vertices and h holes, and let $k \geq 1$ be any integer constant. For any constant $\varepsilon > 0$, we can build a geodesic $(2k + \varepsilon)$-spanner of size $O(c_{\varepsilon,k}\sqrt{h} n \log^2 n)$ and complexity $O(c_{\varepsilon,k}\sqrt{h}(mn^{1/k} + n \log^2 n))$ in $O(c_{\varepsilon,k}\sqrt{h} n \log^2 n + m \log m + K)$ time, where $c_{\varepsilon,k}$ is a constant depending only on ε and k, and K is the output complexity.*

References

1 Mohammad Ali Abam, Marjan Adeli, Hamid Homapour, and Pooya Zafar Asadollahpoor. Geometric spanners for points inside a polygonal domain. In *31st International Symposium on Computational Geometry, SoCG*, volume 34 of *LIPIcs*, pages 186–197, 2015. doi:10.4230/LIPIcs.SOCG.2015.186.

2 Mohammad Ali Abam, Mark de Berg, Mohammad Farshi, Joachim Gudmundsson, and Michiel H. M. Smid. Geometric spanners for weighted point sets. *Algorithmica*, 61(1):207–225, 2011.

3 Mohammad Ali Abam, Mark de Berg, and Mohammad Javad Rezaei Seraji. Geodesic spanners for points on a polyhedral terrain. *SIAM J. Comput.*, 48(6):1796–1810, 2019. doi:10.1137/18M119358X.

4 Sunil Arya, Gautam Das, David M. Mount, Jeffrey S. Salowe, and Michiel H. M. Smid. Euclidean spanners: short, thin, and lanky. In *In Twenty-Seventh Annual ACM Symposium on Theory of Computing, STOC, Proceedings*, pages 489–498. ACM, 1995.

5 Michael A. Bender and Martin Farach-Colton. The LCA problem revisited. In *In Theoretical Informatics, 4th Latin American Symposium, LATIN, Proceedings*, volume 1776 of *Lecture Notes in Computer Science*, pages 88–94. Springer, 2000.

6 Sujoy Bhore and Csaba D. Tóth. Euclidean Steiner spanners: Light and sparse. *SIAM Journal on Discrete Mathematics*, 36(3):2411–2444, 2022.

7 Prosenjit Bose, Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, and Anil Maheshwari. Polygon cutting: Revisited. In *Discrete and Computational Geometry, Japanese Conference, JCDCG, Revised Papers*, volume 1763 of *LNCS*, pages 81–92, 1998. doi:10.1007/978-3-540-46515-7_7.

8 Prosenjit Bose, Rolf Fagerberg, André van Renssen, and Sander Verdonschot. On plane constrained bounded-degree spanners. *Algorithmica*, 81(4):1392–1415, 2019.

9 Prosenjit Bose, Joachim Gudmundsson, and Michiel H. M. Smid. Constructing plane spanners of bounded degree and low weight. *Algorithmica*, 42(3-4):249–264, 2005.

10 Prosenjit Bose and J. Mark Keil. On the stretch factor of the constrained Delaunay triangulation. In *In the 3rd International Symposium on Voronoi Diagrams in Science and Engineering, ISVD*, pages 25–31. IEEE Computer Society, 2006.

11 Prosenjit Bose and Michiel H. M. Smid. On plane geometric spanners: A survey and open problems. *Comput. Geom.*, 46(7):818–830, 2013. doi:10.1016/j.comgeo.2013.04.002.

12 Prosenjit Bose and André van Renssen. Spanning properties of Yao and Θ-graphs in the presence of constraints. *Int. J. Comput. Geom. Appl.*, 29(2):95–120, 2019.

13 T.-H. Hubert Chan, Anupam Gupta, Bruce M. Maggs, and Shuheng Zhou. On hierarchical routing in doubling metrics. *ACM Trans. Algorithms*, 12(4):55:1–55:22, 2016. doi:10.1145/2915183.

14 T.-H. Hubert Chan, Mingfei Li, Li Ning, and Shay Solomon. New doubling spanners: Better and simpler. *SIAM J. Comput.*, 44(1):37–53, 2015.

15 Bernard Chazelle. Triangulating a simple polygon in linear time. *Discret. Comput. Geom.*, 6:485–524, 1991.

16 Kenneth L. Clarkson. Approximation algorithms for shortest path motion planning. In Alfred V. Aho, editor, *In the 19th Annual ACM Symposium on Theory of Computing, STOC*, pages 56–65. ACM, 1987.

17 Gautam Das. The visibility graph contains a bounded-degree spanner. In *In the 9th Canadian Conference on Computational Geometry, CCCG*, 1997.

18 Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational geometry: Algorithms and applications, 3rd Edition*. Springer, 2008. URL: https://www.worldcat.org/oclc/227584184.

19 Sarita de Berg, Marc van Kreveld, and Frank Staals. The complexity of geodesic spanners. *CoRR*, abs/2303.02997, 2023. URL: https://arxiv.org/abs/2303.02997.

20 Michael Elkin and Shay Solomon. Optimal Euclidean spanners: Really short, thin, and lanky. *J. ACM*, 62(5):35:1–35:45, 2015.

21 Lee-Ad Gottlieb and Liam Roditty. An optimal dynamic spanner for doubling metric spaces. In *16th Annual European Symposium on Algorithms, ESA*, volume 5193 of *Lecture Notes in Computer Science*, pages 478–489, 2008. doi:10.1007/978-3-540-87744-8_40.

22 Leonidas J. Guibas and John Hershberger. Optimal shortest path queries in a simple polygon. *J. Comput. Syst. Sci.*, 39(2):126–152, 1989.

23 Leonidas J. Guibas, John Hershberger, Daniel Leven, Micha Sharir, and Robert Endre Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2:209–233, 1987.

24 Sariel Har-Peled and Manor Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM J. Comput.*, 35(5):1148–1184, 2006. doi:10.1137/S0097539704446281.

25 John Hershberger and Subhash Suri. An optimal algorithm for Euclidean shortest paths in the plane. *SIAM J. Comput.*, 28(6):2215–2256, 1999.

26 David G. Kirkpatrick. Optimal search in planar subdivisions. *SIAM J. Comput.*, 12(1):28–35, 1983.

27 Hung Le and Shay Solomon. Truly optimal Euclidean spanners. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 1078–1100. IEEE Computer Society, 2019.

28 Christos Levcopoulos, Giri Narasimhan, and Michiel H. M. Smid. Efficient algorithms for constructing fault-tolerant geometric spanners. In *In the 13th Annual ACM Symposium on the Theory of Computing, STOC, Proceedings*, pages 186–195. ACM, 1998.

29 Joseph S. B. Mitchell and Wolfgang Mulzer. Proximity algorithms. In *Handbook of Discrete and Computational Geometry (3rd Edition)*, chapter 32, pages 849–874. Chapman & Hall/CRC, 2017.

30 Giri Narasimhan and Michiel H. M. Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007.