A Grid-Based Approximation Algorithm for the Minimum Weight Triangulation Problem

Mariëtte Christine Wessels

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mathematics

Sharath Raghvendra
Ezra A. Brown, Chair
William J. Floyd

May 4, 2017
Blacksburg, Virginia

Keywords: Geometric Optimization, Approximation Algorithms, Minimum Weight Triangulation

A Grid-Based Approximation Algorithm for the Minimum Weight Triangulation Problem

Mariëtte C. Wessels

(ABSTRACT)

Given a set of n points on a plane, in the *Minimum Weight Triangulation* problem, we wish to find a triangulation that minimizes the sum of Euclidean lengths of its edges. The problem has been studied for more than four decades and is known to be incredibly challenging. In fact, the complexity status of this problem remained open until recently when it was shown to be NP-Hard. We present a novel polynomial-time algorithm that computes a 16-approximation of the minimum weight triangulation—a constant that is significantly smaller than what has been previously known.

To construct our candidate solution, our algorithm uses grids to partition edges into levels by increasing weights, so that edges with similar weights appear in the same level. We incrementally triangulate the point set by constructing a growing partial triangulation for each level, introducing edges in increasing order of level. At each level, we use a variant of the ring heuristic followed by a greedy heuristic to add edges, finally resulting in a complete triangulation of the point set. In our analysis, we reduce the problem of comparing the weight of the candidate and the optimal solutions to a comparison between the cardinality of the two underlying graphs. We develop a new technique to compare the cardinality of planar straight-line graphs, and in combination with properties due to the imposed grid structure, we bound the approximation ratio.

This work is supported in part by the National Science Foundation under grant NSF-CCF 1464276. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

A Grid-Based Approximation Algorithm for the Minimum Weight Triangulation Problem

Mariëtte C. Wessels

(GENERAL AUDIENCE ABSTRACT)

Given a set of n points on a plane P, a triangulation of P is a set of edges such that no two edges intersect at a point not in P, and the edges subdivide the convex hull of P into triangles. Triangulations have a variety of applications, including computer graphics, finite element analysis, and interpolation, which motivates the need for efficient algorithms to compute triangulations with desirable qualities. The *Minimum Weight Triangulation* problem is the problem of computing the triangulation $\mathcal{T}$ that minimizes the sum of Euclidean lengths of its edges and performs well in many of the above-mentioned applications. The problem has been studied for more than four decades and is known to be incredibly challenging. In fact, the complexity status of this problem remained open until recently when it was shown to be NP-Hard. We present a novel polynomial-time algorithm that computes a 16-approximation of the minimum weight triangulation—a constant that is significantly smaller than what has been previously known. The algorithm makes use of grids together with a variant of the ring and greedy heuristic adapted to apply in a new setting, resulting in an elegant, efficient algorithm.

This work is supported in part by the National Science Foundation under grant NSF-CCF 1464276. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Soli Deo gloria

Acknowledgments

I would like to take this opportunity to thank those that helped me to complete this work:

My advisor, Dr. Sharath Raghvendra, who was willing to take me on as a student, supplied the main ideas for this work, and patiently guided and supported me as we developed, improved upon, and refined these ideas until we were able to present them in this form. I am grateful for the opportunity to learn from him, and the introduction to the remarkable field of computational geometry.

My other committee members: Dr. Floyd, whose careful reading and thoughtful comments greatly improved the quality of this work; Dr. Brown who was willing to serve on my committee in his last semester and brought with him the admiration (shared by many mathematicians) for the magic of Euler's formula.

My excellent fellow office-inhabitants, for obligingly entertaining questions which led to enriching discussions, and sometimes providing unlooked-for rest (by something otherwise known as "distraction").

My parents, who have made many sacrifices in order to ensure that I receive the education they seem to think I deserve, and who managed to instill in me a love and appreciation of learning from a young age.

My beloved sisters, with whom I can share some of the burdens, joys, sorrows, and triumphs of graduate school.

My soon-to-be husband, Joe, who had to endure the brunt of tears and frustration, kept me laughing, encouraged me with his gentle kindness and love, and supported me in this endeavour even though it meant we were nearly 3000 miles apart.

Contents

List of Figures

Chapter 1

Introduction

Consider a planar point set $P \subset \mathbb{R}^2$ with $|P| = n$. A triangulation $\mathcal{T}$ of P is a subdivision of the interior of the convex hull of P into triangles by non-intersecting straight-line segments. Alternatively, any triangulation can be viewed as a maximal planar straight-line graph (PSLG) of the planar point set P. We define the *weight* of an edge as the Euclidean distance between its endpoints, and the weight of any triangulation $\mathcal{T}$ to be the sum of the weights of all the edges in $\mathcal{T}$, denoted $w(\mathcal{T})$. The *Minimum Weight Triangulation* (MWT) problem seeks a triangulation that achieves the minimum weight for P. Note that the MWT for a given point set is not necessarily unique.

1.1 Overview of the MWT Problem

The problem of computing a triangulation for a given point set arises naturally in applications ranging from computer graphics and cartography to finite element meshes and spatial data analysis. For any planar point set, there exist exponentially many possible triangulations. However, all such triangulations do not have equally useful properties, where "useful" is determined by the particular application. Therefore, different applications call for different notions of optimality, many of which have been studied and were surveyed by Bern and Eppstein [2]. For instance, finite element methods and methods using a triangulation for interpolation benefit from a triangulation that avoids long thin triangles. For this reason, the well-known Delaunay triangulation, the dual of the Voronoi diagram, is a popular choice in many applications. This is because the Delaunay triangulation maximizes the minimum angle in the triangulation together with some other optimization criteria. Another notion of optimality is expressed by the MWT problem which yields good results for similar applications, the difference being that the MWT is considerably harder to compute than the Delaunay triangulation. The origin of the MWT problem dates back to 1970 in cartography where it was first considered by Düppe and Gottschalk [5] who originally proposed a greedy

approach to produce a MWT. Shamos and Hoey [16] conjectured that the Delaunay triangulation might be a MWT. However, Lloyd [10] provided counterexamples in 1977 proving that neither the greedy nor the Delaunay triangulation is a MWT in general. At this point the hardness of this problem was unknown. This problem became notorious when Garey and Johnson [7] included the MWT problem in their famous list of twelve major problems with unknown complexity status in 1979. It was not until 2006 when Mulzer and Rote [13] proved that the MWT problem is NP-hard.

1.1.1 Approximation Algorithms

In the light of the MWT problem's being NP-hard, the best one could hope for is a tractible approximation algorithm. An *approximation algorithm* computes a feasible solution for the problem whose objective function is close to the optimal. In this context, "close" means that the value of the objective function for the feasible solution is within a guaranteed factor of the optimal. Suppose $\mathcal{T}^*$ is an optimal MWT for the planar point set P so that $w(\mathcal{T}^*) \leq w(\mathcal{T})$, where $\mathcal{T}$ is any other triangulation on P. If any feasible solution $\mathcal{A}$ generated by an approximation algorithm is such that $w(\mathcal{A}) \leq \alpha \cdot w(\mathcal{T}^*)$, then we say $\mathcal{A}$ is an α-approximation and the algorithm is an α-approximation algorithm. The factor α is known as the *approximation ratio*, since $\frac{w(\mathcal{A})}{w(\mathcal{T}^*)} \leq \alpha$.

Given an approximation algorithm, it is natural to consider whether or not such an algorithm is a good approximation algorithm. Is it possible to improve the approximation algorithm either in terms of a better approximation ratio or a better running time? This leads to the classification of problems in terms of their approximability—bounding the best possible approximation ratio that can be achieved by any approximation algorithm for the problem [8]. An α-approximation algorithm is considered good if α is a constant, and therefore not dependent on the input size n. An improvement upon a constant approximation ratio is when one can construct an *approximation scheme*, where the algorithm yields a $(1+\varepsilon)$-approximation in polynomial time for any $\varepsilon > 0$, yet the running time is allowed to depend on $\frac{1}{\varepsilon}$. Therefore, an approximation scheme can construct a solution for which the approximation ratio is arbitrarily close to 1. One can make the further distinction between a *fully polynomial time approximation scheme* (FPTAS) and a *polynomial time approximation scheme* (PTAS). An algorithm describes a FPTAS provided the running time of the algorithm is polynomial both in the inputs size n and the reciprocal of the error bound $\frac{1}{\varepsilon}$. If the running time is only polynomial in n, the algorithm provides a PTAS. Some problems are known not to have a FPTAS, unless NP = P. The MWT is one such problem. This follows as a consequence of Mulzer and Rote's [13] showing that it is NP-hard to approximate a MWT within a factor of $\mathcal{O}(\frac{1}{n^2})$. Thus, the best possible approximation algorithm for the MWT would be PTAS.

1.1.2 Approximation Algorithms for the MWT Problem

Despite the complexity of the MWT problem remaining unknown for nearly three decades, there were several efforts to design approximation algorithms. This was likely due to an initial lack of progress towards either a polynomial time algorithm for an exact solution, or towards proving the complexity status. The greedy triangulation and Delaunay triangulation were shown to be a factor $\Theta(\sqrt{n})$ [11], and a factor $\Omega(n)$ [11, 9] approximation, respectively, making neither option satisfactory in most applications. Plaisted and Hong [14] described how to approximate the MWT within a factor of $\mathcal{O}(\log n)$ in $\mathcal{O}(n^2 \log n)$ time. This is achieved by partitioning the point set into convex polygons and then repeatedly connecting all pairs of adjacent even numbered vertices. This procedure is known as the *ring heuristic*. The ring heuristic has the virtue of being very simple and easy to implement, but is limited by the fact that it is only defined for convex polygons. Clarkson [3] extended the ring heuristic to non-convex polygons to design an algorithm for the related *Minimum Weight Steiner Triangulation* (MWST) problem (where the MWST differs from the MWT problem in that one may introduce additional input points, *Steiner* points, in the construction of the solution). Levcopoulos and Krznaric [9] proved that a variation of the greedy algorithm approximates the MWT within a very large constant, providing the first constant approximation algorithm for the MWT. This constant was reduced by Yousefi and Young [17], who also discuss the relationship between the MWT and integer linear programs, a relationship first noted by Dantzig *et al.* [4]. About two decades ago, in a major breakthrough, Arora [1] introduced a shifted quadtree-based approach to compute a $(1+\varepsilon)$-approximation for the Euclidean TSP problem. Concurrent to this, but independently, Mitchell [12] also introduced a PTAS for the TSP problem. Arora's technique extended to several other geometric optimization problems. However, as noted by him, the MWT problem resists this approach. While the MWST problem appeared to be amenable to Arora's approach, no approximation scheme has yet been found for the MWST. Finally, Remy and Steger [15] have discovered a quasi-polynomial time approximation scheme (QPTAS) where, for any fixed ε, it yields a $(1+\varepsilon)$-approximation of the MWT in $n^{\mathcal{O}(\log^8 n)}$ time.

1.2 Our Approach and Results

We present a novel algorithm that computes a 16-approximation of the MWT. In comparison, the previous methods for computing a constant approximation [9, 17] achieve an approximation ratio estimated to be higher than 3000. We utilise a grid-based approach [6, 15] combined with a variant of the ring heuristic [14, 3] and a greedy approach [9] to compute our approximate triangulation. More specifically, we maintain a sequence of nested grids (similar to a quad-tree) $G_1, \ldots, G_k$ for some k. Let G_1 be the finest grid and G_k be a single square that contains all input points. We then partition all the $\Theta(n^2)$ edges into levels—an edge appears in level i if and only if its two end points are in neighboring cells in

G_i (note that longer edges have a higher level). We process edges in an increasing order of their levels and compute a maximal PSLG for each level. At each level, this graph partially triangulates the point set. The PSLG is computed by applying a grid-driven ring heuristic that triangulates the regions between alternating reflex chains and convex vertices of polygons using level $i + 1$ edges. We can prove a constant approximation ratio by ensuring that this PSLG generated by our algorithm at level $i + 1$ has a greater number of edges than the number of level i edges of the optimal MWT. Therefore, we ensure that our solution is staying ahead of the optimal triangulation. There are three major technical contributions that assist us in the proof:

- We identify several important properties of a grid-based maximal PSLG of each level. This enables us to show that edges in our partial triangulation at level $i+1$ triangulate a larger region than the level i edges of the optimal triangulation. However, this triangulated region in the restricted optimal triangulation is not necessarily contained inside a triangulated region of our partial triangulation. If this was the case, comparing the cardinality of our solution and the optimal would not be difficult, however since there is no containment, the comparison becomes difficult.

- Nevertheless, we develop a novel technique to relate the cardinality of our partial solution and a restricted optimal triangulation. This technique involves adding edges (not necessarily straight-line) to the restricted optimal triangulation in such a way that for every non-triangulated face of our triangulation, there is a unique non-triangulated face with a greater number of edges in the augmented restricted optimal triangulation. Using Euler's formula, we can then relate the cardinality of our candidate solution and the optimal to achieve a 24-approximation algorithm.

- We choose the side-length of the cells of the grid randomly to improve the expected approximation ratio to 16.

In the following chapters, we present our algorithm and the proof of its correctness. In Chapter 2 we present the preliminary definitions of the grids, edges and non-triangulated faces that are necessary to present our algorithm. The algorithm is described in Chapter 3, and we present and use two algorithmic invariants to bound the approximation ratio in Section 3.2. We prove that the invariants hold in Chapter 4. We conclude in Chapter 5.

Chapter 2

Preliminaries

Let $P \in \mathbb{R}^2$ be the set of n input points. For simplicity of presentation, we will assume that P has a bounded *spread*, i.e., the ratio of the diameter of P and closest pair of points in P is bounded by Δ, where Δ is a power of three. We also scale and translate P so that the closest pair of points in P are at a distance 1, the diameter of P is bounded by Δ, and all points of P are enclosed inside an axis parallel $3\Delta \times 3\Delta$ square S with $(0, 0)$ and $(3\Delta, 3\Delta)$ being the diagonally opposite corners of S. Note that the translation and scaling does not affect the optimal triangulation. We also assume that the points in P are in general position and therefore no three points in P are co-linear. Our algorithm extends to any arbitrary point set that is not in general position and that does not have a bounded spread. However, making these assumptions simplifies the presentation of our algorithm significantly.

For any pair of points, $u, v \in \mathbb{R}^2$, let $\overline{uv}$ be the open straight-line segment in $\mathbb{R}^2$ connecting the points u and v (so the endpoints u and v are not included in $\overline{uv}$). Consider an arbitrary graph $\mathcal{G}$ with the set P as its vertex set. We denote an edge between two vertices in $\mathcal{G}$, $u, v \in P$, as uv. Unless otherwise noted, let $\mathcal{G}$ denote the set of edges of the graph $\mathcal{G}$, so that $|\mathcal{G}|$ denotes the number of edges in $\mathcal{G}$. The edges uv and xy *intersect* if $\overline{uv} \cap \overline{xy} \neq \emptyset$. The graph $\mathcal{G}$ is a *planar straight-line graph* (PSLG) if, for any two edges uv and xy in $\mathcal{G}$, uv and xy do not intersect. For any graph $\mathcal{G}$ (not necessarily planar), a subgraph $\mathcal{M}$ is a *maximal* PSLG, if $\mathcal{M}$ is a PSLG, and for every $uv \in \mathcal{G} \setminus \mathcal{M}$, there exists $u'v' \in \mathcal{M}$ such that uv intersects $u'v'$. It is well-known that any maximal PSLG of the complete graph on P is a triangulation of P with $3n - 3 - h$ edges, where h is the number of points of P that appear on its convex hull. A *maximum* PSLG $\mathcal{M}^* \subset \mathcal{G}$ is a maximal PSLG with the largest number of edges out of all possible maximal PSLGs of $\mathcal{G}$. Note that in general, neither the maximum nor the maximal PSLG is unique.

2.1 Grids and Adjacency Graphs

Our algorithm is described on graphs that are induced by a sequence of nested grids constructed as follows. First, choose a $\gamma \in \mathbb{R}$ uniformly at random from the open interval $(\frac{1}{3}, 1)$. All points in P lie inside the square with diagonal corners $(0,0)$ and $(9\gamma\Delta, 9\gamma\Delta)$. Next, define a sequence of *grids* $G_{\log_3 9\Delta+1}, \ldots, G_0$, where $G_{\log_3 9\Delta+1}$ is the square S. Given a grid G_{i+1}, we obtain grid G_i by simply splitting every *cell* (square) of the grid into 3×3 equal cells. By this construction, grid G_i will have $3^{\log_3 9\Delta - i+1} \times 3^{\log_3 9\Delta - i+1}$ cells each having a side-length of $\gamma 3^{i-1}$. We refer to i as the *level* of G_i. Without loss of generality, we may assume that, for each G_i, every point $p \in P$ is contained in exactly one cell C of G_i. If this is not the case, then one can shift the input point set (by translating every point in P by the same amount and maintaining the same relative position between each input point) such that no input point lies on a grid line. Clearly this does not affect the weight of any triangulation. It is always possible since P consists of finitely many points. The finest grid G_0 consists of cells with side length $\frac{\gamma}{3} < \frac{1}{3}$.

For every cell $C \in G_i$, we will define the *neighboring cells* $N(C)$ of C to be the set of cells in G_i that share a boundary point with C. Our convention is to include C in $N(C)$ since C shares a boundary with itself (thus $N(C)$ contains at most nine cells). Let $\mathcal{N}(C)$ represent the geometric region formed by taking the union of all cells in the neighborhood of C, i.e., $\mathcal{N}(C) = \bigcup_{C \in N(C)} C$. For notational convenience, for any point $p \in P$ contained in the cell C, define $N(p) = N(C)$ and $\mathcal{N}(p) = \mathcal{N}(C)$. We refer to two cells C and C' of a grid G_i as *diagonal neighbors* if their boundaries share exactly one vertex but do not share any edge. If C and C' share exactly one edge, they are referred to as *orthogonal neighbors*. Any cell can have at most four diagonal neighbors, and four orthogonal neighbors.

For each grid G_i, the *adjacency graph* $\mathcal{G}_i$ connects every pair of distinct points that are in neighboring cells with an edge. Therefore, the vertex set of the adjacency graph $\mathcal{G}_i$ is P itself, and for any two points $u, v \in P$, such that $u \neq v$, uv is an edge in $\mathcal{G}_i$ if and only if $u \in \mathcal{N}(v)$. It is not difficult to see that $\mathcal{G}_0 = \emptyset$, $\mathcal{G}_{\log_3 9\Delta+1}$ is the complete graph on n vertices, and $\mathcal{G}_i \subseteq \mathcal{G}_{i+1}$ for all $0 \leq i < \log_3 9\Delta + 1$. As we go from level $i-1$ to level i in the adjacency graph, several new edges may be added. We refer to these edges as *level i edges*, denoted $\mathcal{S}_i = \mathcal{G}_i \setminus \mathcal{G}_{i-1}$. Note that for any edge $ab \in \mathcal{S}_i$, it follows from the definition of a neighborhood that $\gamma 3^{i-2} \leq \|ab\| \leq 6\sqrt{2}(\gamma 3^{i-2})$.

2.2 Non-triangulated Faces

A straight-line embedding of a planar graph $\mathcal{G}$, defined on the point set P, subdivides the plane into regions. Each region is referred to as a *face* of $\mathcal{G}$. More formally, each face is a maximal connected region in $\mathbb{R}^2 \setminus (P \cup (\bigcup_{uv \in \mathcal{G}} \overline{uv}))$. Since a face is a subset of $\mathbb{R}^2$, the *boundary* of the face can be defined in the standard way (a point x is on the boundary of a face f if,

for every $\varepsilon > 0$, the Euclidean ball centered at x with radius ε contains both a point in f and a point not in f). Therefore, faces are open subsets of $\mathbb{R}^2$ and their boundaries consist of points in P and line segments (which corresponds to edges) of the graph $\mathcal{G}$. We refer to a bounded face f as a *triangulated face* if its boundary is a single connected component with exactly three edges and three vertices of $\mathcal{G}$. Any other face (bounded or unbounded) is *non-triangulated*. The boundary of a non-triangulated face may consist of one or more isolated vertices, connected cycles, or trees. For example, in Figure 2.1b the unbounded face f consists of two polygons and an edge on its boundary.

Let f be a non-triangulated face of $\mathcal{G}$. Our algorithm applies a grid-based modified ring heuristic on a clockwise ordering of the vertices for each connected component of the boundary of f. We generate this clockwise ordering as a sequence $\sigma(f)$ of the vertices that appear on the boundary of f by walking along the boundary so that the face f appears on the right. Let $\overrightarrow{uv}$ denote walking along any edge uv from u to v. During this construction, any edge $\overrightarrow{uv}$ has been explored if we have already walked from u to v along the edge uv. If uv has the same face on both its sides, then we will explore $\overrightarrow{uv}$ and $\overrightarrow{vu}$ as we generate the sequence for that face, which will result in a single vertex appearing multiple times in the boundary vertex sequence as part of distinct directed "explorations". Otherwise, the edge is explored only once for this sequence (with an orientation so that f appears on the right). We construct the vertex sequence $\sigma(f)$ as follows: start the sequence with an arbitrary vertex v_0 on the boundary. If the connected component is an isolated vertex, then $\sigma(f) = \langle v_0 \rangle$ is the vertex sequence. Otherwise, let x be any vertex adjacent to v_0 that also appears on the boundary of f. If f appears on the right as we walk along any edge $\overrightarrow{v_0x}$, set $v_1 \leftarrow x$, the second vertex in the sequence. To determine the rest of the sequence, suppose we have generated the first i vertices $\sigma(f) = \langle v_0, \ldots, v_{i-1} \rangle$ of the sequence. To generate the $(i+1)^{\text{th}}$ vertex v_i in the sequence, we choose the potential next vertex as follows:

- if v_{i-1} is a degree 1 vertex in $\mathcal{G}$, let x be the only adjacent vertex;
- if v_{i-1} had degree greater than 1, consider the ray r passing through v_{i-2} and v_{i-1}. Let x be the first vertex adjacent to x that r intersects when r is rotated about v_{i-1} in the anti-clockwise direction.

If $\overrightarrow{v_{i-1}x}$ has not been explored, we walk along $\overrightarrow{v_{i-1}x}$ from v_{i-1} to x, set $v_i \leftarrow x$, and add v_i to $\sigma(f)$. We repeat the same process to find the next vertex in the sequence. Otherwise, if $\overrightarrow{v_{i-1}x}$ has already been explored, we return $\sigma(f)$ as the clockwise ordering of the vertices. By repeating this process for every connected component of the boundary, we generate a separate clockwise ordering of vertices for each connected component of the boundary of f (for instance, two boundary vertex sequences are generated for f_0 in Figure 2.1a). Note that any vertex may appear multiple times in $\sigma(f)$. In Figure 2.1b, x_1 appears multiple times in $\sigma(f) = \langle v_0, v_1, \ldots, v_{10} \rangle$ but labeled as v_1, v_4, and v_{10} depending on where in the sequence the vertex was encountered. In each of these instances, x_1 appears between two different vertices in the clockwise ordering of the boundary, and therefore generates a distinct element

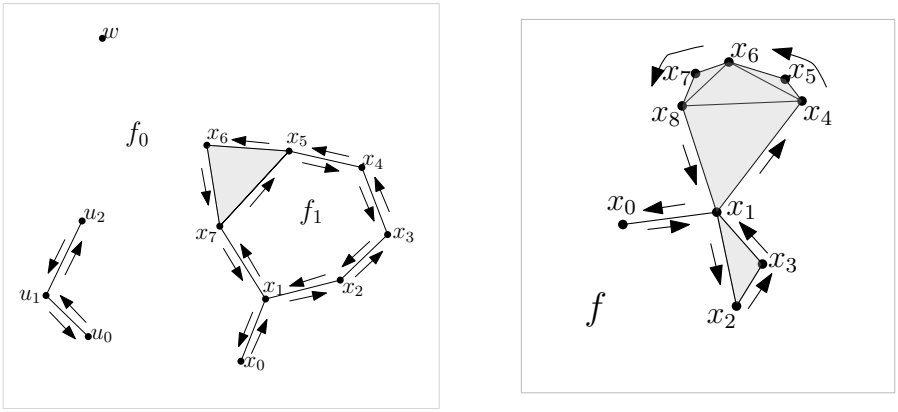

Figure 2.1: (a) f_0 has a disconnected boundary, and will therefore have more than one vertex sequence $\sigma_u(f_0) = \langle u_0, u_1, u_2, u_1\rangle$, and $\sigma_v(f_0) = \langle x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_1\rangle$. The other non-triangulated face f_1 has only one vertex sequence $\sigma(f_1) = \langle x_1, x_7, x_5, x_4, x_3, x_2\rangle$. (b) the vertex sequence constructed for f is $\sigma(f) = \langle x_0, x_1, x_2, x_3, x_1, x_4, x_5, x_6, x_7, x_8, x_1\rangle$. Note that shaded faces are triangulated.

in $\sigma(f)$. For $\sigma(f) = \langle v_0, \ldots, v_{m-1}\rangle$, and for any two integers $0 \leq i, k \leq m-1$, let v_{i+k} denote the vertex v_j where $j = i + k \pmod m$. Note that one can also define a boundary vertex sequence consisting of exactly three vertices for any triangulated face of $\mathcal{G}$.

Suppose $\sigma(f) = \langle v_0, \ldots, v_{m-1}\rangle$ is a vertex sequence for a non-triangulated face f of $\mathcal{G}$, and let v_i be any vertex in the sequence. We say that v_i is a *convex vertex* if, as you walk from v_{i-1} through v_i to v_{i+1}, we make a right turn at v_i. Otherwise, if we make a left turn, we refer to v_i as a *reflex vertex* (since we assume no three points are co-linear, every vertex is either reflex or convex). For any $0 \leq i < j \leq m-1$, we refer to the contiguous subsequence $\langle v_i, \ldots, v_j\rangle$ as a *chain* from v_i to v_j and denote it by $\mathbb{C}(v_i, v_j)$ with $|\mathbb{C}(v_i, v_j)| = j - i$. The *interior* of the chain is the set of all vertices in the chain except the first and last vertices v_i and v_j. A *reflex chain* is a chain consisting only of reflex vertices. We refer to the reflex chain from v_i to v_j as a *maximal* reflex chain if v_{i-1} and v_{j+1} are not reflex.

Let $\mathcal{G}$ be a PSLG, and f be a non-triangulated face of $\mathcal{G}$ with the boundary vertex sequence $\sigma(f) = \langle v_0, \ldots, v_i, \ldots, v_j, \ldots, v_{m-1}\rangle$. The vertex v_j is *visible* to a vertex v_i in $\sigma(f)$ if $\overline{v_i v_j} \subset f$ and adding the edge $v_i v_j$ to $\mathcal{G}$ creates two faces f' and f'' with $\sigma(f') = \langle v_i, v_{i+1} \ldots, v_{j-1}, v_j\rangle$ and $\sigma(f'') = \langle v_i, v_j, v_{j+1}, \ldots, v_{i-1}\rangle$. Note that the definition of visibility is symmetric, i.e., for $x, y \in \sigma(f)$, x is visible to y if and only if y is visible to x. We say that v_i and v_j are δ-visible if v_i is visible to v_j and the length of $\overline{v_i v_j}$ is at most δ. We extend this definition and define the visibility between a vertex and any point on a line segment. For any $v_i \in \sigma(f)$, we say that a point $q \in \overline{v_j v_{j+1}}$ is visible to a vertex v_i if v_i is visible to q in the vertex sequence $\sigma(f) = \langle v_0, \ldots, v_i, \ldots, v_j, q, v_{j+1}, \ldots, v_{m-1}\rangle$. We say that a vertex v_i is visible to the edge $v_j v_{j+1}$ if there exists a point $q \in \overline{v_j v_{j+1}}$ with the property that v_i is visible to q; if in addition

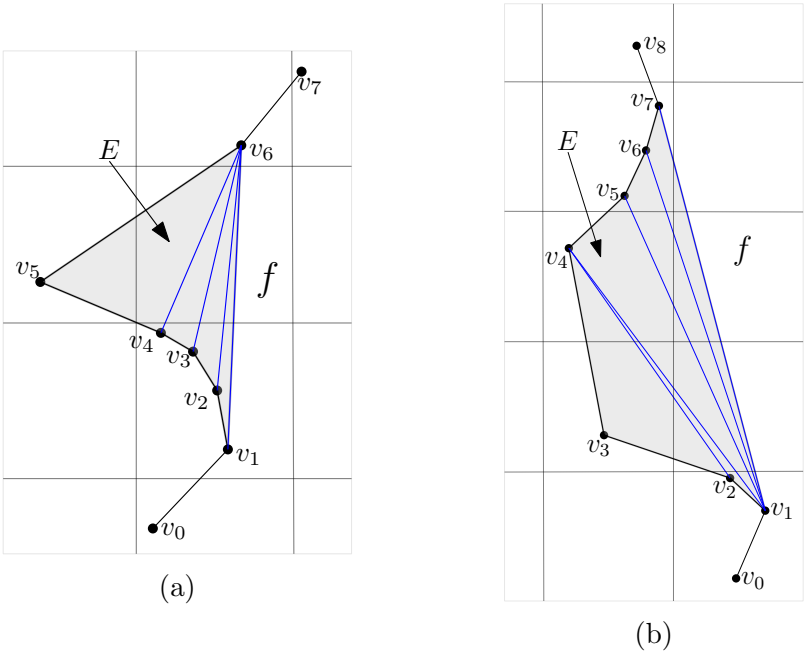

Figure 2.2: (a) $\mathbb{C}(v_1, v_6)$ is a 1-chain, and (b) $\mathbb{C}(v_1, v_7)$ is a 2-chain, that can be triangulated with blue edges.

one can choose q such that the length of $\overline{v_i q}$ is at most δ, v_i is δ-visible to $v_j v_{j+1}$.

Let $\sigma(f) = \langle v_0, v_1, \ldots v_{m-1} \rangle$ be the boundary vertex sequence for some non-triangulated face in a PSLG. For any vertex $v_j \in \sigma(f)$, let k be the smallest integer such that $k > j$ and $v_{k(\text{mod } m)}$ is convex. We define v_k to be the *forward convex vertex* of v_j. Also, define v_{k+1} to be the *forward support vertex* of v_j denoted by $\text{fwd}(v_j, \sigma(f))$. For any reflex vertex v_j in $\sigma(f)$, let k be the largest integer such that $k < j$ and $v_{k(\text{mod } m)}$ is convex. We define v_k to be the *backward convex vertex* of v_j. Then we define v_{k-1} to be the *backward support vertex* of v_j denoted by $\text{back}(v_j, \sigma(f))$.

Non-triangulated face and the grid. We will show that in the first part of the algorithm, while executing the modified ring heuristic on any non-triangulated face f of a maximal PSLG $\mathcal{A}_i$, the algorithm generates two particular types of chains. We refer to these as *type* 1-*chains* and *type* 2-*chains*. Let $\sigma(f) = \langle v_0, v_2, \ldots, v_{m-1} \rangle$ be the boundary vertex sequence of f that is being processed. A chain from v_i to v_k is a type 1-*chain*, provided (i) it has exactly one convex vertex v_j for $i < j < k$, (ii) every vertex participating in the chain is contained in $\mathcal{N}(v_j)$, (iii) every vertex from v_{j+1} to v_k is visible to v_{j-1} and symmetrically, every vertex from v_i to v_{j-1} is visible to v_{j+1} in $\sigma(f)$. A chain from v_i to v_j is a type 2-*chain* provided (i) it has exactly two consecutive convex vertices, v_j and v_{j+1} such that $i + 1 < j + 1 < k$, and (ii) the chain from v_i to v_{j+1} is a type 1-chain and the chain from v_j to v_k is a type 1-chain (see Figure 2.2). For brevity, we refer to these as 1- and 2-chains.

Let $\mathbb{C}(v_i, v_j)$ be a 1-chain or a 2-chain. Define $E(v_i, v_j)$ to be the region bounded by $\mathbb{C}(v_i, v_j)$ and the segment $\overline{v_i v_j}$ in $\sigma(f)$ (Figure 2.2). The effectiveness of the algorithm partially rests on the fact that $E(v_i, v_j)$ can be triangulated in a straight-forward fashion, provided it does not have any other input points in its interior. In Figure 2.2a, $\mathbb{C}(v_1, v_6)$ is a 1-chain with v_5 the only convex vertex in the chain, and the region $E = E(v_1, v_6)$ can easily be triangulated. In Figure 2.2b, $\mathbb{C}(v_1, v_7)$ is a 2-chain with v_3 and v_4 the two consecutive convex vertices, and the region $E = E(v_1, v_7)$ can also be triangulated as shown.

In the following chapter, we present our algorithm to compute an approximate Minimum Weight triangulation.

Chapter 3

Algorithm

3.1 Algorithm

The algorithm iteratively constructs a candidate PSLG $\mathcal{A}_i$ which is a subgraph of $\mathcal{G}_i$ for each grid level i. Each $\mathcal{A}_i$ can be seen as a partial triangulation of P. Initialize $\mathcal{A}_0 \leftarrow \emptyset$. We construct $\mathcal{A}_{i+1}$ from $\mathcal{A}_i$ in two phases. In the first phase, we process every face f of $\mathcal{A}_i$ in the clockwise order, compute particular chains, and triangulate their regions by adding some of the edges of $\mathcal{S}_{i+1}$ which results in an intermediate PSLG $\hat{\mathcal{A}}_i$. Chains generated by the algorithm are denoted $\mathbb{C}(v_k)$, omitting the last vertex in the chain, and where the chain is determined by the start vertex. In the second phase, we process all the edges of $\mathcal{S}_{i+1}$ that have not yet been added in a greedy fashion to construct a maximal PSLG $\mathcal{A}_{i+1} \subset \mathcal{G}_{i+1}$. Given $\mathcal{A}_i$, the partial triangulation for the next level, $\mathcal{A}_{i+1}$, is computed from $\mathcal{A}_i$ as follows.

Phase 1: Initialize $\hat{\mathcal{A}}_i \leftarrow \mathcal{A}_i$. We describe our algorithm for each non-triangulated face f of $\mathcal{A}_i$. If the boundary of f contains more than one connected component, we repeat the algorithm for the boundary vertex sequence of each connected component of f. Let $\sigma(f) = \langle v_0, \ldots, v_{m-1} \rangle$ be any such vertex sequence to be processed. We begin by choosing a start vertex, and then proceed to describe how each vertex is processed.

Choosing a start vertex. Choose the starting vertex v_j to be any backward support vertex in $\sigma(f)$, i.e., $v_j = \text{back}(v', \sigma(f))$ for some $v' \in \sigma(f)$. If there is no such vertex then let $v_j = v_0$.

Processing $\sigma(f)$. Starting from v_j, we will *visit* all the vertices as they appear in $\sigma(f)$. If an edge incident on vertex v is added, then v is *processed*. Initially $k \leftarrow 0$ and $f' \leftarrow f$. As we visit the vertices, we add new edges, each of which splits f into a triangle and a smaller non-triangulated face f'. We will dynamically maintain the vertex sequence $\sigma(f')$.

We process v_{j+k} by repeating (1) and (2) until $k > m - 1$.

1. For v_{j+k}, let v_{l-1} be its forward convex vertex and let v_l be its forward support vertex in $\sigma(f)$. Mark v_{j+k} as visited. If v_l is visible to v_{j+k} in f and $v_{l-1} \in \mathcal{N}(v_{j+k})$ then set the chain $\mathbb{C}(v_{j+k}) \leftarrow \mathbb{C}(v_{j+k}, v_l)$ (Figure 3.1a). Else set $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_{j+k})$ (the trivial chain with only one vertex).

   If v_{l-1} is not visited and $|\mathbb{C}(v_{j+k})| > 1$, then

   (a) If v_{l+1} is reflex, let v_{q+1} be its backward convex vertex and v_q be its backward support vertex. If $v_{q+1} \in \mathcal{N}(v_{l+1})$ and v_q is visible to v_{l+1} in f, then

   - Let $\mathbb{C}(v_{l+1}, v_r)$ be the maximal reflex chain that starts at v_{l+1}. Scan along this chain and identify the last vertex v_s for which $v_{q+1} \in \mathcal{N}(v_s)$ and v_q is visible to v_s in f. Set $\mathbb{C}(v_{j+k}) \leftarrow \mathbb{C}(v_{j+k}, v_s)$ (Figure 3.1b).

   (b) Now add an edge from v_l to every vertex in $\mathbb{C}(v_{j+k}, v_{l-2})$. If $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_s)$, then we add edges from v_{j+k} to every vertex in $\mathbb{C}(v_{l+1}, v_s)$. Let $\mathcal{E}(v_{j+k})$ be the set of edges added. Set $k \leftarrow |\mathcal{E}(v_{j+k})| + 1$. Mark every vertex in the chain $\mathbb{C}(v_{j+k})$ as visited and update f' by removing all vertices in the interior of $\mathbb{C}(v_{j+k})$ from $\sigma(f')$ (see Figure 3.1).

   Otherwise if v_{l-1} is visited, then

   (c) Let $v_{j'}$ be the vertex appearing immediately after v_j in $\sigma(f')$. If $v_{j'}$ is visible to v_{j+k} in $\sigma(f')$ set the chain $\mathbb{C}(v_{j+k}) \leftarrow \mathbb{C}(v_{j+k}, v_{j'})$ (Figure 3.1c). Add an edge from $v_{j'}$ to every vertex in $\mathbb{C}(v_{j+k}, v_{l-2})$. Mark every vertex in the chain $\mathbb{C}(v_{j+k})$ as visited and update f' by removing all the vertices in the interior of $\mathbb{C}(v_{j+k})$ from $\sigma(f')$. Set $k \leftarrow m$.

2. If no edges are added in step (1), then set $k \leftarrow k + 1$.

Phase 2: For each edge e in $\mathcal{S}_{i+1}$ that has not yet been added to $\hat{\mathcal{A}}_i$, if $\hat{\mathcal{A}}_i \cup e$ is still a planar graph, add e to $\hat{\mathcal{A}}_i$. Once all edges of $\mathcal{S}_{i+1}$ are processed, the resulting graph is a maximal PSLG $\mathcal{A}_{i+1} \subset \mathcal{G}_{i+1}$.

These two phases are repeated for each level i, until $\mathcal{A}_{\log_3 9\Delta+1}$, a maximal PSLG on the complete graph of the input set (and hence a triangulation), is computed, which yields our approximate solution. Note that the number of non-empty levels for the edges can not exceed $O(n^2)$ and for each level, the processing time is polynomial. Therefore, the execution time of our algorithm is a polynomial independent of Δ.

While processing a vertex v_t, suppose the algorithm executes step 1(c). Note that this case only arises when the forward convex vertex of v_t (v_{l-1} for that step) is the start vertex v_j or the following vertex v_{j+1}. Since this step sets $k = m$, the algorithm finishes Phase 1 for the face f after it finishes 1(c). Alternatively, if the forward support vertex v_l of v_t is the start vertex, then processing for that sequence will terminate without executing 1(c). Call v_t, the

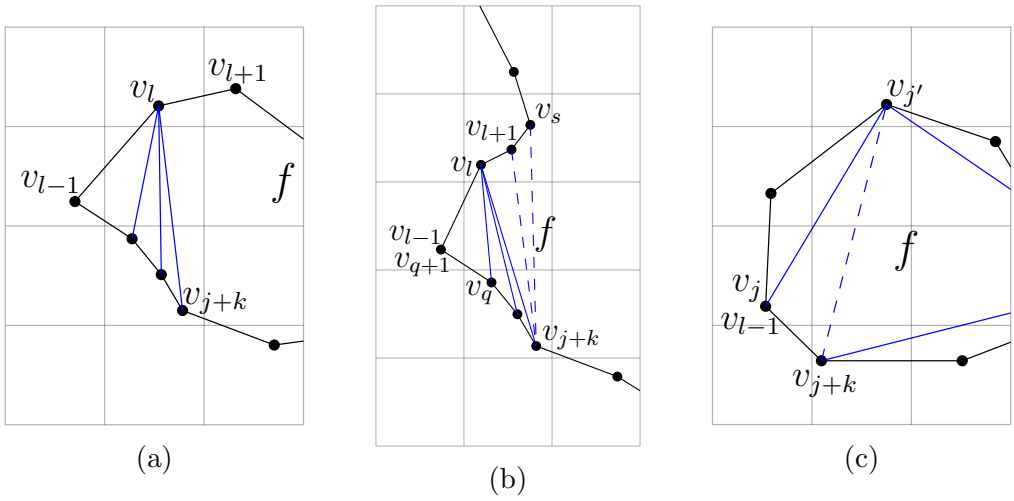

Figure 3.1: Edges added in Phase 1 (in blue). (a) $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_l)$ and v_{l+1} is not reflex. (b) v_{l+1} is reflex, so step 1(a) is executed and $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_s)$. (c) v_{l-1} has already been visited and $v_{l-1} = v_j$, so $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_{j'})$.

last vertex to be processed in that sequence, the end vertex. The following observations follow from the description of the algorithm:

(O1) For any pair of vertices v, v' processed by the algorithm, the chains $\mathbb{C}(v)$ and $\mathbb{C}(v')$ are interior disjoint except for the pair corresponding to the start vertex $\mathbb{C}(v_j)$ and the end vertex $\mathbb{C}(v_t)$. For v_j and v_t, either $\mathbb{C}(v_j)$ is contained inside $\mathbb{C}(v_t)$ (if step 1(c) was executed) or $\mathbb{C}(v_j)$ and $\mathbb{C}(v_t)$ are interior disjoint (v_j was the forward support of v_t).

(O2) The last vertex $v_{k'}$ on any maximal reflex chain $\mathbb{C}(v_k, v_{k'})$ will have an edge to its forward support v_l provided the forward convex vertex $v_{l-1} \in \mathcal{N}(v_{k'})$ and the vertex v_l is visible to $v_{k'}$.

3.2 Approximation Ratio

In this section, we begin by showing that the algorithm presented in Section 3.1 produces, in the worst case, a 24-approximation of the MWT. We will also show that the expected approximation ratio of our algorithm is 16. Our algorithm maintains two invariants. To state the invariants, we first introduce the *disc graph* $\mathcal{D}_i$, defined as follows. Let P be the vertex set of $\mathcal{D}_i$, and given two vertices $u, v \in P$, $uv \in \mathcal{D}_i$ if and only if $\|uv\| \leq \frac{\gamma 3^{i-1}}{\sqrt{2}}$. In other words, the point v is contained in a disc that is centered at u with a radius of $\frac{\gamma 3^{i-1}}{\sqrt{2}}$. Clearly $\mathcal{D}_i \subseteq \mathcal{G}_i$ for all i. Furthermore, let $\mathcal{T}^*$ be the optimal triangulation of P, and let $\mathcal{T}_i^*$ be the edges of the optimal triangulation that are in the disc graph $\mathcal{D}_i$, i.e., $\mathcal{T}_i^* = \mathcal{T}^* \cap \mathcal{D}_i$.

With these definitions, we can state the invariant (divided into two parts) that the algorithm maintains for every level $i \in \{0, \ldots, \log_3 9\Delta + 1\}$.

Invariant 1. *$\hat{\mathcal{A}}_i$ is a* PSLG *and every edge $uv \in \hat{\mathcal{A}}_i \setminus \mathcal{A}_i$ has length $\|uv\| \leq 4\sqrt{2} \cdot 3^{i-1}$.*

Invariant 2. $|\hat{\mathcal{A}}_i| \geq |\mathcal{T}_i^*|$.

Note that the length bound in Invariant 1 for edges added in Phase 1 of the algorithm implies that all of the edges added are in $\mathcal{S}_{i+1}$. Assuming both invariants hold, we will bound the approximation ratio of our algorithm.

Theorem 1 (Approximation Ratio)**.** *Suppose Invariants 1 and 2 hold. Then the candidate solution produced by the algorithm in Section 3.1, $\mathcal{A}_{\log_3 9\Delta+1}$, is an α-approximate MWT, where $\alpha \leq 24$. Furthermore, the expected value of α is at most* 16*.*

Proof. Suppose that both the invariants hold. Let $\mathcal{A} = \mathcal{A}_{\log_3 9\Delta+1}$ be the triangulation computed by our algorithm. Any triangulation for a given point set P has $m = 3n - 3 - h$ edges, where h is the number of points on the convex hull of P. Therefore, $|\mathcal{A}| = |\mathcal{T}^*| = m$. Let $\tau = \langle a_1, \ldots, a_m \rangle$ be the ordering of edges of $\mathcal{A}$ based on when they were added by the algorithm, i.e., edge a_i appears before edge a_j in τ if a_i was added before a_j by the algorithm. Note that in the sequence τ, the edges of level i appear before all level $i+1$ edges. Let $\mathcal{K}_i = \mathcal{T}_i^* \setminus \mathcal{T}_{i-1}^*$. We also order the edges of the optimal triangulation $\mathcal{T}^*$ into another sequence $\tau^* = \langle t_1, \ldots, t_m \rangle$. In this sequence, for any $i < j$, the edges of $\mathcal{K}_i$ appear before the edges of $\mathcal{K}_j$. Let t_j be the j^{th} edge in τ^*, and a_j be the j^{th} edge of τ. To prove this theorem, we will show that $\alpha_j = \frac{w(a_j)}{w(t_j)} \leq 24$ and $\mathbb{E}[\alpha_j] \leq 16$. Given this, we obtain the bound on α as follows:

$$\alpha = \frac{w(\mathcal{A})}{w(\mathcal{T}^*)} \quad = \quad \sum_{j=1}^{m} \frac{w(t_j)}{w(\mathcal{T}^*)} \cdot \frac{w(a_j)}{w(t_j)} = \sum_{j=1}^{m} \beta_j \cdot \frac{w(a_j)}{w(t_j)}, \tag{3.1}$$

where $\beta_j = \frac{w(t_j)}{w(\mathcal{T}^*)}$. Since $\beta_j > 0$ and $\sum_{j=1}^{m} \beta_j = 1$, α is a weighted average of all α_j values. Therefore, we can bound α and $\mathbb{E}[\alpha]$ by providing an upper bound for α_j and $\mathbb{E}[\alpha_j]$.

From here on, we will bound α_j by 24 and $\mathbb{E}[\alpha_j]$ by 16 to prove the theorem. Let k be an integer such that $t_j \in \mathcal{K}_k$. Therefore, the cost of t_j, $w(t_j) \geq \frac{\gamma 3^{k-2}}{\sqrt{2}}$. By Invariant 2, it follows that $j \leq |\hat{\mathcal{A}}_k|$, so $a_j \in \hat{\mathcal{A}}_k$. By Invariant 1, $w(a_j) \leq \gamma 4\sqrt{2} \cdot 3^{k-1}$. Therefore,

$$\frac{w(a_j)}{w(t_j)} \leq \frac{\gamma 4\sqrt{2} \cdot 3^{k-1}}{\frac{\gamma 3^{k-2}}{\sqrt{2}}} = 24$$

Thus, $\mathcal{A}$ is an 24-approximation of $\mathcal{T}^*$.

Recollect that γ is chosen uniformly at random from the interval $(\frac{1}{3}, 1)$. α_j may be expressed as a function of γ as follows. Let i be the largest integer such that $w(t_j) \geq \frac{\gamma 3^{i-1}}{\sqrt{2}}$. Let t_j be

the edge pq in $\mathcal{T}^*$. By the choice of i, $\frac{\gamma}{3} \leq \frac{\sqrt{2}\cdot\|pq\|}{3^i} \leq \gamma$. Since $\gamma \in (\frac{1}{3}, 1)$, either $t_j \in \mathcal{K}_{i+1}$ or $t_j \in \mathcal{K}_{i+2}$. Hence there are two possible cases:

- If $\gamma \in (\frac{1}{3}, \frac{\sqrt{2}\|pq\|}{3^i})$, then $t_j \in \mathcal{K}_{i+2}$. From Invariant 2, $w(a_j)$ will be at most $4\sqrt{2}(\gamma 3^{i+1})$. Therefore α_j is $\frac{4\sqrt{2}\gamma 3^{i+1}}{\|pq\|}$.
- If $\gamma \in [\frac{\sqrt{2}\|pq\|}{3^j}, 1)$ then $t_j \in \mathcal{K}_{i+1}$. From Invariant 2, $w(a_j)$ will be at most $4\sqrt{2}(\gamma 3^i)$. Therefore α_j is $\frac{4\sqrt{2}\gamma 3^i}{\|pq\|}$

Let $x = \frac{\sqrt{2}\|pq\|}{3^i}$. The expected value of α_j can be expressed as:

$$\begin{aligned}\mathbb{E}[\alpha_j] &= \frac{3}{2}\int_{1/3}^{x}\left(\frac{4\sqrt{2}\gamma 3^{i+1}}{\|pq\|}\right)\mathrm{d}\gamma + \frac{3}{2}\int_{x}^{1}\left(\frac{4\sqrt{2}\gamma 3^{i}}{\|pq\|}\right)\mathrm{d}\gamma \\ &= \frac{2\sqrt{2}3^{i+1}}{\|pq\|}\left(x^2 + \frac{1}{3}\right) = \frac{4\sqrt{2}\|pq\|}{3^{i-1}} + \frac{2\sqrt{2}3^i}{\|pq\|} \leq 16\end{aligned}$$

The last inequality holds for any p, q such that $\|pq\| \in \left[\frac{3^{i-1}}{\sqrt{2}}, \frac{3^i}{\sqrt{2}}\right]$, and hence for every t_j being considered. □

Chapter 4

Proof of Algorithmic Invariants

To complete the proof of Theorem 1, we prove Invariant 1 in Section 4.2 and Invariant 2 in Section 4.4, which were assumed in the proof of the approximation ratio. It will be useful to first consider properties of a maximal PSLG with respect to $\mathcal{G}_i$ which arise due to the underlying grid structure. These properties can then be used to prove Invariant 1.

4.1 Properties of a Maximal PSLG in $\mathcal{G}_i$

In order to state the properties satisfied by any maximal PSLG of $\mathcal{G}_i$, we introduce the following definitions. For any two points u and v, let C_u and C_v be the cells of G_i that contain u and v. Let $\Gamma_i^0(u) = \{C_u\}$ and let $\Gamma_i^j(u) = \bigcup_{C \in \Gamma_i^{j-1}} N(C)$. We say that u and v are k *cells apart* if $C_v \in \Gamma_i^{k-1}$. It is not difficult to see that if u and v are k cells apart, then $\|uv\| \leq k\sqrt{2} \cdot \gamma 3^{i-1}$ and the worst case is achieved when u and v are diagonally opposite corners of the square of side length $k\gamma 3^{i-1}$ containing k^2 cells of G_i.

Next, we state the properties. The proofs of each of these properties are deferred until Section 4.5. Let $\mathcal{A}$ be a maximal PSLG with respect to $\mathcal{G}_i$. Let f be a non-triangulated face of $\mathcal{A}$ with a boundary vertex sequence $\sigma(f) = \langle v_0, \ldots, v_{m-1} \rangle$.

(P1) Suppose $v_j \in \sigma(f)$ is convex. Then v_{j-1}, v_j, and v_{j+1} are in three distinct cells $C_{v_{j-1}}$, C_{v_j}, and $C_{v_{j+1}}$ of G_i, respectively, and $C_{v_{j-1}} \notin N(C_{v_{j+1}})$.

(P2) Suppose $v_j \in \sigma(f)$ and $v_k v_{k+1}$ is any edge on the boundary of f such that $v_k \in \mathcal{N}(v_j)$ (resp. $v_{k+1} \in \mathcal{N}(v_j)$) and $v_k v_{k+1}$ is visible to v_j in $\sigma(f)$. Then,

(i) the chain $\mathbb{C}$ from v_j to v_{k+1} (resp. chain $\tilde{\mathbb{C}}$ from v_k to v_j) in $\sigma(f)$ is a 1-chain, and,

(ii) v_{k+1} is a forward (resp. v_k is a backward) support vertex for every vertex from v_j to v_{k-1} (resp. from v_{k+2} to v_j).

(P3) For any chain $\mathbb{C}(v, y)$ from v to y in $\sigma(f)$,

(i) if $\mathbb{C}(v, y)$ is a 1-chain with v' as its only convex vertex, then the region $E = E(v, y)$ is contained in $\mathcal{N}(v')$, i.e., $E \subset \mathcal{N}(v')$, and E contains no input points of P, other than those in $\mathbb{C}(v, y)$.

(ii) if the chain $\mathbb{C}(u, y)$ is a 2-chain with v and x as the two convex vertices, then the region $E = E(u, y)$ is such that $(E \cap (\mathcal{N}(v) \cup \mathcal{N}(x))) \cap P$ contains only points of $\mathbb{C}(u, y)$. In only two cases, $E \not\subset (\mathcal{N}(v) \cup \mathcal{N}(x))$ (See Figure 4.1). In all other cases, E contains only points of $\mathbb{C}(u, y)$, but no other points of P.

(P4) For any vertex $v \in \sigma(f)$, if an edge xy on the boundary of f is δ-visible for $\delta = \frac{3^{i-1}}{\sqrt{2}}$, then exactly one of x and y are in $\mathcal{N}(v)$ and the other is not.

4.2 Proof of Invariant 1

Using (P1)–(P4), we now establish Invariant 1. To assist with presentation, we use the notation used in the description of the algorithm. Recollect that while processing a vertex v_{j+k}, v_{l-1} and v_l are its forward convex vertex and forward support vertex, respectively. If v_{l+1} is reflex, then v_{q+1} and v_q are the backward convex vertex and the backward support vertex, respectively. For every edge uv added in Phase 1, let $\mathbb{C}(u, v)$ in $\sigma(f)$ be the chain from u to v. We refer to this kind of chain $\mathbb{C}(u, v)$ as a *triangulated chain*. If uv appears on the boundary of a non-triangulated face of $\hat{\mathcal{A}}_i$ after Phase 1 has been executed, then $\mathbb{C}(u, v)$ is a *maximal* triangulated chain. If the chain $\mathbb{C}(u, v)$ was generated when processing u, we may omit the the other endpoint of the chain, and simply denote the chain by $\mathbb{C}(u)$.

Invariant 1 asserts two claims, (i) any edge $uv \in \hat{\mathcal{A}}_i \setminus \mathcal{A}_i$ has length $\|uv\| \leq 4\sqrt{2} \cdot \gamma 3^{i-1}$, and (ii) that the intermediate graph $\hat{\mathcal{A}}_i$ produced by Phase 1 is a PSLG. Note that to prove the first claim (i), it is sufficient to show that the endpoint of any edge uv that was added by the algorithm is at most 4 cells apart. We first show that any two vertices in a 1-chain are at most 3 cells apart and any two vertices in a 2-chain are at most four cells apart (Lemma 2). We then show that any triangulated chain generated by the algorithm is either a 1-chain or a 2-chain (Lemma 3). This proves (i). We show (ii) that planarity is never violated by edges added in Phase 1 in Lemma 4 and 5. It follows that the algorithm triangulates the region E of each triangulated chain and therefore that the algorithm maintains Invariant 1.

Lemma 2. *If u, v are vertices in a 1-chain, then u and v are at most 3 cells apart. If u, v are vertices in a 2-chain, then u and v are at most 4 cells apart.*

Proof. Suppose u and v are two vertices of a 1-chain $\mathcal{C}$. Let C_u and C_v be the cells of G_i containing u and v, respectively. Let v' be the only convex vertex of $\mathcal{C}$. By definition of 1-chain, $u, v \in \mathcal{N}(v')$, therefore $u \in \mathcal{N}(v')$ and $v' \in \mathcal{N}(v)$. So $C_u \in \Gamma_i^2(v)$, therefore, by definition u and v are at most 3 cells apart.

Suppose u and v are vertices in a 2-chain with v' and v'' as the two convex vertices. It follows from the definition of a 2-chain that $u, v \in \mathcal{N}(v') \cup \mathcal{N}(v'')$ and $v' \in \mathcal{N}(v'')$. Therefore, any two cells of $\mathcal{N}(v') \cup \mathcal{N}(v'')$ are at most 4-cells apart and it follows that u and v are at most 4 cells apart. □

We now show that the triangulated chains generated by the algorithm are 1- or 2-chains.

Lemma 3. *Every maximal triangulated chain $\mathbb{C}(v_{j+k})$ is either a 1-chain or a 2-chain. Furthermore, the maximal triangulated chain $\mathbb{C}(v_j)$ for the start vertex is a 1-chain.*

Proof. Note that for any vertex to be processed, step 1 of the algorithm must be executed. First, show that the maximal triangulated chain $\mathbb{C}(v_j)$ of the starting vertex is a 1-chain. The start vertex v_j is chosen to be the backward support vertex, unless none exists. Suppose v_j is a backward support vertex for some vertex v', so $v_j = \text{back}(v', \sigma(f))$. By definition, the vertex that appears after v_j, v_{j+1}, is a convex vertex (the backward convex vertex of v') and v_{j+2} is a reflex vertex. However, it is also follows from the definition that $v_{j+1} = v_{l-1}$ is the forward convex vertex of v_j and $v_{j+2} = v_l = \text{fwd}(v_j, \sigma(f))$ is the forward support vertex of v_j. If $v_{j+1} \in \mathcal{N}(v_j)$ and v_{j+2} is visible to v_j then $\mathbb{C}(v_j) = \mathbb{C}(v_j, v_{j+2})$ (otherwise no edges are added and we are done). v_j visible to v_{j+2} implies that $v_{j+1}v_{j+2}$ is visible so by (P2), the chain $\mathbb{C}(v_j, v_l)$ is a 1-chain. If step 1(a) is not executed, then we are done. Suppose step 1(a) is executed, so $\mathbb{C}(v_j) = \mathbb{C}(v_j, v_s)$, where v_s is the last reflex vertex on the reflex chain in which v_{j+2} appears, such that v_j is visible to v_s and $v_s \in \mathcal{N}(v_{j+1})$. Since all vertices from v_{j+3} to v_s are reflex, it follows that $\mathbb{C}(v_j)$ is a 1-chain. Therefore, in either case $\mathbb{C}(v_j)$, the chain generated for the start vertex, must be a 1-chain. Now suppose $v_j = v_0$ so there are no reflex vertices in $\sigma(f)$. Then v_{j+1} is the forward convex vertex and v_{j+2} is the forward support that is also convex. It follows that v_{j+2} is visible to v_j, $\mathbb{C}(v_j)$ has only one convex vertex in its interior, and $v_j, v_{j+2} \in \mathcal{N}(v_{j+1})$. Thus $\mathbb{C}(v_j)$ is a 1-chain.

Next, we prove that the claim holds for v_t, the last vertex that is processed for the particular $\sigma(f)$. Either step 1(c) is executed when processing v_t or it is not. Suppose step 1(c) is executed when processing v_t to generate $\mathbb{C}(v_t)$; since this is a special case we prove this separately. If step 1(c) is being executed, the forward convex vertex v_{l-1} is already visited when v_t is being processed. If $v_{l-1} \in \mathcal{N}(v_t)$ and $v_{j'}$ is visible to v_t the maximal triangulated chain is $\mathbb{C}(v_t, v_{j'})$ (if $v_{l-1} \notin \mathcal{N}(v_t)$ or $v_{j'}$ is not visible to v_t no edges will be added and we are done). Note that v_{j+1} is convex (due to the choice of starting vertex), therefore, the forward convex vertex is either $v_{l-1} = v_j$ or $v_{l-1} = v_{j+1}$. If $v_{l-1} = v_j$, then v_j and v_{j+1} are both convex. It follows from (P2) that $\mathbb{C}(v_t, v_{j+1})$ is a 1-chain, and by the above argument $\mathbb{C}(v_j) = \mathbb{C}(v_j, v_{j'})$ is also a 1-chain. Therefore, by definition $\mathbb{C}(v_t) = \mathbb{C}(v_t, v_{j'})$ is a 2-chain.

If $v_{j+1} = v_{l-1}$, then v_j is reflex. We have already shown that $\mathbb{C}(v_j) = \mathbb{C}(v_j, v_{j'})$ must be a 1-chain, therefore $\mathbb{C}(v_j)$ contains only one convex vertex in its interior, and thus the chain $\mathbb{C}(v_t, v_{j'})$ also contains only one convex vertex, namely v_{j+1}, in its interior. Since $v_t \in v_{j+1}$ it follows that $\mathbb{C}(v_j, v_{j'})$ is a 1-chain. The remaining case for v_t, when step 1(c) is not executed, uses the same argument as for any other v_{j+k}; the argument for this case is provided next.

Finally, we prove that for any other vertex v_{j+k}, which is not the starting vertex, and is the ending vertex only if step 1(c) is not executed, the maximal triangulated chain $\mathbb{C}(v_{j+k})$ is either a 1-chain or 2-chain. Therefore step 1(c) is not executed. If $v_{l-1} \notin \mathcal{N}(v_{j+k})$ or v_l is not visible to v_{j+k}, then no edges are added and we are done. So suppose $v_{l-1} \in \mathcal{N}(v_{j+k})$ and v_l is visible to v_{j+k}. This implies that the edge $v_{l-1}v_l$ is visible to v_{j+k}, so by (P2) the chain $\mathbb{C}(v_{j+k}, v_l)$ generated by the algorithm with v_{l-1} as the only convex vertex is a 1-chain. If step 1(a) is not executed we are done. Suppose step 1(a) is executed, then if $v_{q+1} \in \mathcal{N}(v_{l+1})$ and v_q is visible to v_{l+1} (if these conditions are not met, then we are done), we can use the same argument to show that $\mathbb{C}(v_q, v_{l+1})$ is a 1-chain. Since all vertices from v_{l+1} to v_s are reflex, such that $v_{q+1} \in \mathcal{N}(v_{l+1})$ and v_q is visible, by extension it follows that $\mathbb{C}(v_{l+1}, v_s)$ is also a 1-chain. Now there are two possibilities: either (i) $v_{q+1} = v_{l-1}$ or (ii) $v_{q+1} = v_l$. In case (i), the chain $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_s)$ has only one convex vertex, $v_{q+1} = v_{l-1}$ and by construction every vertex of this chain is inside $\mathcal{N}(v_{q+1})$, so that $\mathbb{C}(v_{j+k})$ is a 1-chain. Otherwise, in case (ii), v_l and v_{l-1} are adjacent convex vertices; it then follows that $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_s)$ is a 2-chain. This completes the proof. □

Lemma 3 implies that the algorithm only generates 1-chains and 2-chains in Phase 1, therefore by applying Lemma 2 it follows that any edge added by the algorithm in Phase 1 has length at most $4\sqrt{2} \cdot \gamma 3^{i-1}$. Note that a useful observation in light of this proof is that the algorithm only generates 2-chains if it executes step 1(a) or 1(c) of the algorithm. Furthermore, for a triangulated chain $\mathbb{C}(u, v)$, since $\mathbb{C}(u, v)$ is either a 1-chain or a 2-chain, it follows that the edges added for this chain $\mathcal{E}(u, v)$ will not intersect with each other. In order to establish that $\hat{\mathcal{A}}_i$ is planar, we show that edges added for two distinct chains do not intersect, nor do they intersect with any edges in $\mathcal{A}_i$.

Lemma 4. *Let $\mathbb{C}(u, y)$ be the maximal triangulated chain with v and x as the two convex vertices in its interior. Then, the region $E(u, y)$ bounded by $\mathbb{C}(u, y)$ and the line segment $\overline{uy}$ contains only input points also in $\mathbb{C}(u, y)$ and no other points of P.*

Proof. For the sake of contradiction, let us assume that there is a vertex $u \in \sigma(f)$ such that when u is processed, the algorithm generates a maximal triangulated chain $\mathbb{C}(u, y)$ and the region $E(u, y)$ contains at least one input point $q \in P$ which is not a vertex in $\mathbb{C}(u, y)$. By property (P3), if $\mathbb{C}(u, y)$ is a 1-chain, then $E(u, y)$ contains no vertex $q \notin \mathbb{C}(u, y)$. Therefore, since we assume there is such a q, $\mathbb{C}(u, y)$ must be a 2-chain. By Lemma 3, $\mathbb{C}(v_j)$ is a 1-chain, therefore u cannot be the start vertex v_j. Since $\mathbb{C}(u, y)$ is a 2-chain, it follows by Property (P3) that $\mathcal{N}(v) \cup \mathcal{N}(x)$ does not contain q such that $\overline{uy} \not\subset \mathcal{N}(v) \cup \mathcal{N}(x)$ only if $\mathbb{C}(u, y)$ is in one of the two possible configurations. Figure 4.1 depicts these two possible configurations

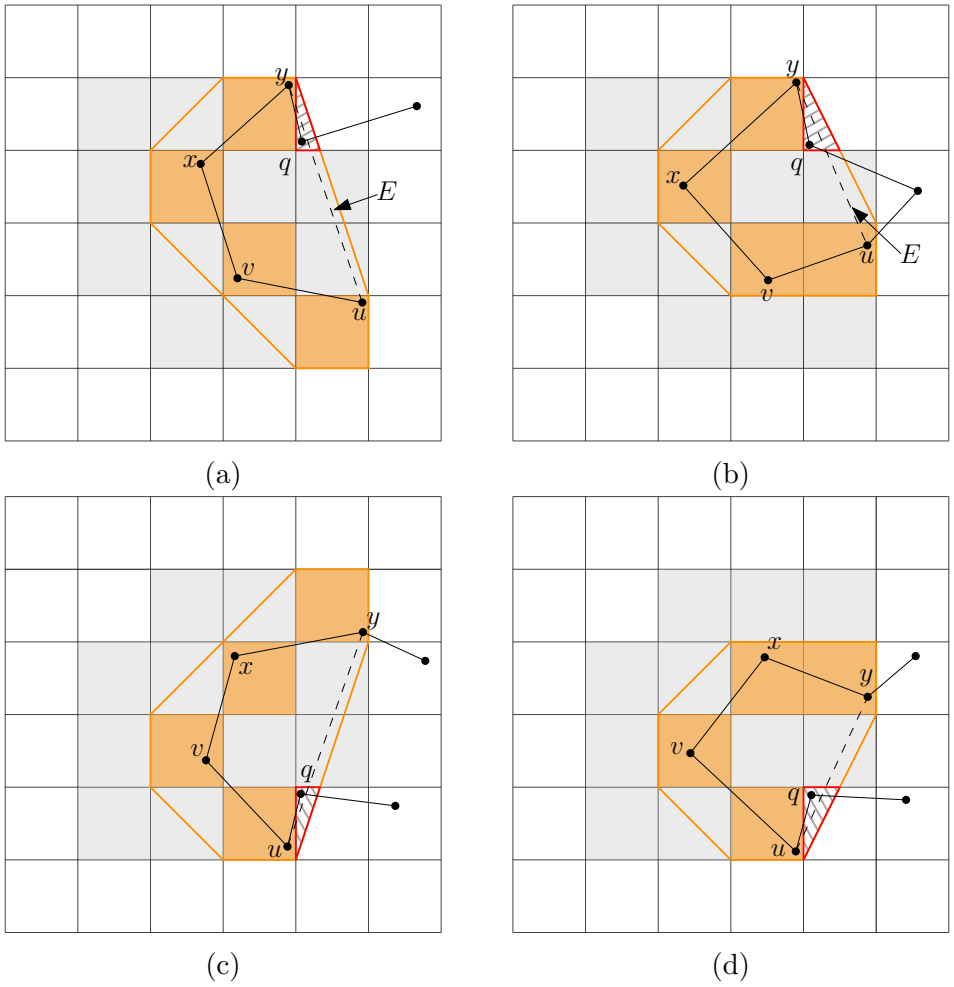

Figure 4.1: Exceptions in (P3): two possible configurations depicted by (a) and (b) of four cells such that $E(u, y)$ is not contained in $\mathcal{N}(v) \cup \mathcal{N}(x)$ ($N(v) \cup N(x)$ is depicted in grey). The vertex $q \in E(u, y)$ but q is not in $\mathbb{C}(u, y)$. (c) and (d) are mirror images of (a) and (b), respectively.

through four representative examples (two possibilities for each of the two cases). We will show that the vertex q is a reflex vertex in each of these configurations which implies that the algorithm would not have processed u, only visited, resulting in a contradiction. Recall that the algorithm only generates a 2-chain if it enters step 1(a) or 1(c) while processing u. Presenting the arguments for the representative cases in Figure 4.1 is sufficient, since reflex vertices between u and v or x and y do not affect the argument.

Consider case (a) and (b) in Figure 4.1. Here $q \in \mathcal{N}(y)$. Let C_q be the cell in G_i that contains q. If q is convex, the vertex immediately following q in $\sigma(f)$, call it q', lies in E so that either $q' \in C_q$ or $q' \in \mathcal{N}(v) \cup \mathcal{N}(x)$. If $q' \in \mathcal{N}(v) \cup \mathcal{N}(x)$ the region between x, y, and q' would be triangulated so that x and y do not appear on the boundary, so $q' \in C_q$. If q is reflex it follows that q' is visible to y, so the edge $yq' \in \mathcal{A}_i$. Therefore there must exist a reflex vertex immediately after y in $\sigma(f)$. So without loss of generality, let q be that vertex. By assumption v is the forward convex vertex of u, and v is convex, so $\mathbb{C}(u, y)$ could not have been generated in step 1(a). If $\mathbb{C}(u, y)$ was generated in step 1(c), then v was already visited by the algorithm and v was the start vertex, however, this is also impossible, since v is not a backwards support of any vertex. However, x is a backward support vertex, so x should have been the start vertex. Therefore, (a) and (b) are impossible. Now consider the mirror cases (d) and (e) in Figure 4.1. Here $q \in \mathcal{N}(u)$. We can use a similar argument than above to show that a reflex vertex q immediately preceding u in $\sigma(f)$ exists. Since u is convex and not the starting vertex, it follows that q must have been processed by the algorithm by observation (O2), which implies that u would not have been processed by the algorithm because it would be in the interior of $\mathbb{C}(q)$. Therefore, (d) and (e) is also impossible. The result follows. □

Lemma 5. *For any i, $\hat{\mathcal{A}}_i$ is a* PSLG*, after Phase* 1 *of the algorithm has been completed.*

Proof. We will show that while processing a non-triangulated face f, the edges added do not intersect with each other or intersect with any edge of $\mathcal{A}_i$. Given any two maximal triangulated chains $\mathbb{C}(v_k)$ and $\mathbb{C}(v_{k'})$ from (O1), we know that these chains are interior disjoint. This implies that edges in $\mathcal{E}(v_k)$ cannot intersect with edges in $\mathcal{E}(v_{k'})$. From Lemma 4, we know that the regions $E(v_k)$ and $E(v_{k'})$ do not contain any points apart from those in $\mathbb{C}(v_k)$ and $\mathbb{C}(v_{k'})$. This implies that no edges of $\mathcal{A}_i$ (other than those with endpoint in the chain) can intersect $E(v_k)$ or $E(v_{k'})$, since such edges would either violate planarity of $\mathcal{A}_i$ or they would have an endpoint in $E(v_k)$ or $E(v_{k'})$. Therefore, edges added by the algorithm do not intersect with edges already in $\mathcal{A}_i$. It follows that $\hat{\mathcal{A}}_i$ is planar. □

This completes the proof of Invariant 1. In the following section we show that after Phase 1, the intermediate PSLG $\hat{\mathcal{A}}_i$ has a particular property that plays a critical role in proving Invariant 2.

4.3 δ-Visibility in $\hat{\mathcal{A}}_i$

In the following Lemma we prove a critical property used in the proof of Invariant 2, together with some consequences of this property.

Lemma 6. *Let f be a non-triangulated face in $\mathcal{A}_i$ with vertex sequence $\sigma(f)$. Let $\sigma(f') = \langle v_{j_0}, \ldots, v_{j_p} \rangle$ be the resulting vertex sequence after Phase 1 of the algorithm, and let $\delta = \frac{\gamma 3^{i-1}}{\sqrt{2}}$.*

Suppose $v \in \sigma(f)$, and xy is an edge on the boundary of f such that xy is δ-visible to v. Then if $v \in \mathbb{C}(v_{j_r}, v_{j_{r+1}})$, it follows that $x, y \in \mathbb{C}(v_{j_{r-1}}, v_{j_r})$, or $x, y \in \mathbb{C}(v_{j_r}, v_{j_{r+1}})$, or $x, y \in \mathbb{C}(v_{j_{r+1}}, v_{j_{r+2}})$.

Proof. Suppose $v \in \sigma(f)$ and xy is an edge on the boundary of f, so x and y are consecutive vertices in $\sigma(f)$. This lemma claims that v and x, y either belong to the same maximal triangulated chain, or adjacent maximal triangulated chains (here adjacent means the chains share exactly one endpoint in $\sigma(f')$). Without loss of generality, assume as one walks from v in $\sigma(f)$, one encounters x before y. Note that a symmetric argument holds for the case where one encounters x first, followed by y and then v. Suppose, for the sake of contradiction, that v is a in maximal triangulated chain $\mathbb{C}(v_{j_r}, v_{j_{r+1}})$, and x is in the maximal triangulated chain $\mathbb{C}(v_{j_d}, v_{j_{d+1}})$, where $r + 1 < d$ (so v and x are in distinct non-adjacent chains). Note that x and y must belong to the same triangulated chain. This follows from the fact that x and y are adjacent in $\sigma(f)$ so that there are only three possibilities: (i) xy also appears on the boundary of f', (ii) x and y are both in the interior of the same triangulated chain, or (iii) x is an endpoint and y is in the interior of the same chain (or *vice verca*). It is clear that x and y still belong to the same chain in all three cases. Since v and x are not in adjacent (or the same) triangulated chains, there is at least one triangulated chain $\mathbb{C}^*$ between $\mathbb{C}(v_{j_r}, v_{j_{r+1}})$ and $\mathbb{C}(v_{j_d}, v_{j_{d+1}})$, distinct from either. Every triangulated chain is either a 1-chain or a 2-chain (Lemma 3), so $\mathbb{C}^*$ has at least one convex vertex v'. Therefore, as you walk along $\sigma(f)$ from v you encounter v' and then x. By property (P4), $x \in \mathcal{N}(v)$, since xy is δ-visible to v. So one can apply (P2) and it follows that x must be convex. This contradicts the fact that by (P1) the vertices before v' and after v' must belong to different cells that are not in each others' neighborhoods. It follows that there can be no triangulated chain between the triangulated chain of v and the triangulated chain of x. Hence the maximal triangulated chain containing v is either the same chain that contains x and y, or is adjacent to the maximal triangulated chain containing x and y, as claimed. □

The next lemma applies Lemma 6 to $\mathcal{T}_i^*$, establishing a crucial property that helps us relate the cardinality of $\hat{\mathcal{A}}_i$ to the cardinality of $\mathcal{T}_i^*$ in the following section. This lemma shows that for a non-triangulated face f in $\hat{\mathcal{A}}_i$, if an edge in $uv \in \mathcal{T}_i^*$ intersects the boundary of f, then u and v are not inside f, and as you walk along $\overrightarrow{uv}$, every time you enter f, you leave f on a boundary edge that is adjacent to the one through which you entered. As a corollary, it follows that distinct connected components of $\hat{\mathcal{A}}_i$ cannot be arbitrarily close to each other.

Lemma 7. *Fix a non-triangulated face f of $\hat{\mathcal{A}}_i$. For any edge uv of $\mathcal{T}_i^*$ such that u and v are not on the boundary of f, let $\{\overline{a_1b_1}, \ldots, \overline{a_kb_k}\}$ be the open line segments resulting from taking $\overline{uv} \cap f$, where each $\overline{a_jb_j} \subset (\overline{uv} \cap f)$. Then, for each $\overline{a_jb_j}$, there are three consecutive vertices v_{k-1}, v_k and v_{k+1} in $\sigma(f)$ with $a_j \in \overline{v_{k-1}v_k}$ and $b_j \in \overline{v_kv_{k+1}}$, and v_k is convex in $\sigma(f)$.*

Proof. Let f be a non-triangulated face in $\hat{\mathcal{A}}_i$, and let uv be any edge in $\mathcal{T}_i^*$ such that u and v do not appear on the boundary of f. Let $\delta = \frac{\gamma 3^{i-1}}{\sqrt{2}}$. If $\overline{uv} \cap f = \emptyset$, the statement holds trivially. So assume $\overline{uv} \cap f \neq \emptyset$ and $\{\overline{a_1b_1}, \ldots, \overline{a_kb_k}\}$ is a set of line segments formed by $\overline{uv} \cap f$. Consider one such segment $\overline{a_jb_j}$. Then $\overline{a_jb_j}$ intersects two edges on the boundary of f, v_pv_{p+1} and v_qv_{q+1} in $\sigma(f)$. For the sake of contradiction, assume $p+1 \neq q$. By definition, any edge of $\mathcal{T}_i^*$ has a length less than or equal to δ. In particular, $\|uv\| \leq \delta$. Without loss of generality, assume uv is oriented vertically, then one can imagine sliding uv horizontally to the left or to the right. The distance between v_pv_{p+1} and v_qv_{q+1} can increase, at most, in one direction (since we are considering straight-line edges). Assume the distance increases or stays the same to the right, then slide uv to the left (so that the distance stays the same or reduces) until uv intersects an input point x. Such a point exists, because if no other point is encountered, sliding uv in this manner would encounter one of the endpoints of v_pv_{p+1} or v_qv_{q+1}. It follows immediately that x is δ-visible to v_pv_{p+1} and v_qv_{q+1}. By Lemma 6, it follows that these vertices would be part of the same or adjacent maximal triangulated chains in $\mathcal{A}_i$ so that in $\hat{\mathcal{A}}_i$ v_pv_{p+1} and v_qv_{q+1} must either be the same edge, or adjacent on the boundary of f. Since a straight-line embedding of uv intersects f by passing through v_pv_{p+1} and v_qv_{q+1} the only possibility is for $v_{p+1} = v_q$ to be the same convex vertex in $\sigma(f)$, as claimed. □

Corollary 8. *Let f be a non-triangulated face in $\hat{\mathcal{A}}_i$, and uv any edge in $\mathcal{T}_i^*$ such that u and v are not on the boundary of f. If uv intersects any two boundary edges e_1 and e_2 of f, then e_1 and e_2 belong to the same connected component of $\hat{\mathcal{A}}_i$.*

Proof. Suppose f is a non-triangulated face of $\hat{\mathcal{A}}_i$, $uv \in \mathcal{T}_i^*$ such that u and v does not belong to the boundary of f, but uv intersects the boundary of f. Let $\{\overline{a_1b_1}, \ldots, \overline{a_kb_k}\}$ be a set of line segments resulting from $f \cap \overline{uv}$, where each $\overline{a_jb_j} \subset \overline{uv}$. Consider one such segment $\overline{a_jb_j}$. By Lemma 7, $a_j \in \overline{v_{j-1}v_j}$ and $b_j \in \overline{v_jv_{j+1}}$ for three consecutive vertices v_{j-1}, v_j, and v_{j+1} in $\sigma(f)$, and therefore $v_{j-1}v_j$ and v_jv_{j+1} belong to the same component in $\hat{\mathcal{A}}_i$. Since this is the case for each a_jb_j, the result follows. □

Consider a subset $C \subseteq P$ such that vertices in C belong to a single maximal connected component of $\hat{\mathcal{A}}_i$. It follows from Corollary 8, that the input points in C correspond to maximal connected components in $\mathcal{T}_i^*$. Thus, if a vertex $p \in C$, then p cannot belong to the same connected component as any vertex $p' \notin C$. Note that vertices in C may, however, form more than one maximal connected component in $\mathcal{T}_i^*$. This observation allows us to show that Invariant 2 is satisfied for each connected component in $\hat{\mathcal{A}}_i$ and the corresponding connected components of $\mathcal{T}_i^*$ separately in Section 4.4. Since each argument can be applied

to any connected component, it is sufficient to present our argument for one such connected component, and for simplicity, it is convenient to use $\hat{\mathcal{A}}_i$ to denote this component.

4.4 Proof of Invariant 2

Invariant 2 compares the cardinality of the PSLG $\hat{\mathcal{A}}_i$ to the cardinality of the PSLG $\mathcal{T}_i^*$. If the region triangulated by $\hat{\mathcal{A}}_i$ contains the region triangulated by $\mathcal{T}_i^*$, then it is easy to show that $\hat{\mathcal{A}}_i$ has a greater number of edges than $\mathcal{T}_i^*$. However, as shown in Lemma 7, edges of $\mathcal{T}_i^*$ can intersect non-triangulated faces of $\hat{\mathcal{A}}_i$ and therefore, these two PSLGs triangulate different regions. Nevertheless, Lemma 7 implies that edges of $\mathcal{T}_i^*$ which intersect a non-triangulated face of $\hat{\mathcal{A}}_i$ f, will intersect two edges of $\hat{\mathcal{A}}_i$ that are adjacent on the boundary of f. In order to establish Invariant 2, we will show that the regions triangulated by $\mathcal{T}_i^*$ and $\hat{\mathcal{A}}_i$ are "close" (cf. Figure 4.2) to each other. In Section 4.4.1 we provide conditions under which we can compare the cardinalities of any two PSLGs. After that, in Section 4.4.2, we show these conditions are satisfied for $\hat{\mathcal{A}}_i$ and $\mathcal{T}_i^*$, which allows us to prove Invariant 2.

4.4.1 Comparing Cardinality

Let $\mathcal{G}$ be any planar graph. Let F denote the set of faces of this planar embedding of $\mathcal{G}$. When the graph being considered is not clear, we denote the set of faces by $F(\mathcal{G})$. For any face $f \in F$ and its vertex sequence $\sigma(f)$, we define its *signature*, $s(f)$, to be the length of the vertex sequence $\sigma(f)$, i.e., $s(f) = |\sigma(f)|$. Let X be a connected planar graph and Y be any planar graph. For any two faces $f_1 \in F(Y)$ and $f_2 \in F(X)$, we say that f_1 *dominates* f_2 if and only if $s(f_1) \geq s(f_2)$. Suppose, for every non-triangulated face f in $F(X)$, there is a unique dominating face in Y, then we will show that $|X| \geq |Y|$ (Corollary 10). In Section 4.4.2, we will use this to prove Invariant 2.

Lemma 9. *Consider a connected planar graph $\mathcal{G}$ and let F be all the faces in the planar embedding of $\mathcal{G}$. The total number of edges in the graph $|\mathcal{G}|$ can be written as*

$$|\mathcal{G}| = 3n - 6 - \sum_{f \in F}(s(f) - 3)$$

Proof. For any face $f \in F$, its signature is the length of the vertex sequence. Recollect that we construct the vertex sequence by exploring the edges of the boundary of the face f in the clockwise direction. Every edge e has at most two faces, one on each side. We refer to these faces as the co-faces of e.

By the construction of the vertex sequence, every edge contributes 1 to the signature of each of its co-faces. If the edge e has only 1 co-face f, it contributes two to $s(f)$. Therefore,

$\sum_{f\in F} s(f) = 2|\mathcal{G}|$. From Euler's formula, we know $|\mathcal{G}| = n + |F| - 2$ and therefore, $3|\mathcal{G}| = 3n - 6 + 3|F|$. It follows that,

$$|\mathcal{G}| + \sum_{f\in F} s(f) = 3n - 6 + 3|F|, \text{ so } |\mathcal{G}| = 3n - 6 - \sum_{f\in F}(s(f) - 3).$$

□

In Lemma 9, if the graph is disconnected, then we can extend the proof to show that the number of edges is strictly smaller than $3n - 6 - \sum_{f\in F}(s(f) - 3)$. In addition, note that for any triangular face f, $s(f)-3 = 0$. Hence, the total number of edges in any connected planar graph can be calculated using only the size of the vertex sequences of the non-triangulated faces. Using this observation, we obtain the following:

Corollary 10. *Let X be a connected planar graph and Y be any planar graph. For every non-triangulated face $f \in F(X)$, suppose there is a unique non-triangulated face $f' \in F(Y)$ such that $s(f) \leq s(f')$. Then, $|X| \geq |Y|$.*

Proof. Let $F^*(X)$ be the set of non-triangulated faces of X, and $F^*(Y)$ be the set of non-triangulated faces of Y. Applying Lemma 9 and the observation that for every triangular face f of X, $s(f) - 3 = 0$, we can express the number of edges in X as

$$|X| = 3n - 6 - \sum_{f\in F^*(X)} (s(f) - 3).$$

Since, for every face $f \in F^*(X)$ there is a unique face $f' \in F^*(Y)$ such that $s(f') \geq s(f)$ and since $s(f') - 3 \geq 0$, we have $\sum_{f\in F^*(X)}(s(f) - 3) \leq \sum_{f'\in F^*(Y)}(s(f') - 3)$. Since Y is not necessarily connected, using an almost identical argument to Lemma 9, we can express the number of edges in Y by the following inequality,

$$\begin{aligned} |Y| &\leq 3n - 6 - \sum_{f\in F^*(Y)} (s(f) - 3) \\ &\leq 3n - 6 - \sum_{f\in F^*(X)} (s(f) - 3) \\ &= |X|. \end{aligned}$$

□

It follows from Corollary 10, that in order to determine whether one can apply Lemma 9, one need only be concerned with the non-triangulated faces of a particular planar graph. Thus, from this point on, we let $F(X)$ denote only the non-triangulated faces of a planar graph X.

4.4.2 Proving Invariant 2

Our Strategy. To prove Invariant 2, we will add edges to $\mathcal{T}_i^*$ in such a way that $\mathcal{T}_i^*$ dominates $\hat{\mathcal{A}}_i$ in order to apply Corollary 10. Fix any non-triangulated face $f \in F(\hat{\mathcal{A}}_i)$. When we overlay f on the straight line embedding of $\mathcal{T}_i^*$, we will use Lemma 7 to show that f has only one connected "non-trivial" intersection with some non-triangulated face f' of $\mathcal{T}_i^*$. We define the region of interest $f \cap f'$ as the trace of f. Next, we augment the graph $\mathcal{T}_i^*$ by embedding new edges (planar but not necessarily straight-line) and carefully create a new face f'' around the trace of f. We show that f'' dominates f and refer to f'' as the *dominating face* of f. After this procedure is repeated for every non-triangulated face in $F(\hat{\mathcal{A}}_i)$, the augmented graph $\mathcal{T}_i^*$ now dominates $\hat{\mathcal{A}}_i$. However, adding edges to $\mathcal{T}_i^*$ may create duplicate edges, so that $\mathcal{T}_i^*$ a multi-graph and Corollary 10 does not apply. We show that duplicate edges cannot participate in two distinct dominating faces. Removing one of the duplicate edges will merge two faces h and h' and create a new face h'' that has a signature greater than or equal to the signature of h or h'. Since either h or h' (and not both) can be a dominating face of some face $f \in F(\hat{\mathcal{A}}_i)$, h'' will be the unique dominating face of f. After deleting the duplicate edges, we now have an augmented $\mathcal{T}_i^*$ that is a planar graph that dominates $\hat{\mathcal{A}}_i$. One may apply Corollary 10, and Invariant 2 follows. We begin by introducing the definitions that is required to formalize this argument.

Suppose f is a non-triangulated face of $\hat{\mathcal{A}}_i$ with the boundary vertex sequence $\sigma(f)$. Let uv be an edge in $\mathcal{T}_i^*$. We say that an edge uv is a *crossing edge* for any convex vertex $v_j \in \sigma(f)$ if the edge uv intersects the edges $v_{j-1}v_j$ and v_jv_{j+1}. We direct the crossing edge uv from u to v if we first encounter the edge $v_{j-1}v_j$ as we move from u to v. In this case, we refer to v as the head and u as the tail of this directed edge $\overrightarrow{uv}$. Note that any such edge $uv \in \mathcal{T}_i^*$ can be a crossing edge for many convex vertices in $\sigma(f)$, and any convex vertex in $\sigma(f)$ can have zero, one, or many crossing edges. However, from Lemma 7, we know that any edge $uv \in \mathcal{T}_i^*$ that intersects f has to be a crossing edge for some convex vertex $v \in \sigma(f)$. For any convex vertex $v_j \in \sigma(f)$, let $\mathcal{CS}(v_j)$ be the set of crossing edges of v_j. We set $\mathcal{CS}(v_j)$ to be empty if v_j is a reflex vertex. If $\mathcal{CS}(v_j)$ is not empty, we define the furthest crossing edge of v_j as the first edge of $\mathcal{CS}(v_j)$ that we encounter as we walk from v_{j-1} to v_j. We can equivalently define the furthest crossing edge to be the last edge of $\mathcal{CS}(v_j)$ that we encounter as we walk from v_j to v_{j+1}. This follows from the fact that $\mathcal{T}_i^*$ is a planar graph, hence crossing edges cannot intersect. We make the following straight-forward observations for crossing edges which follows form Lemma 7 and the fact that $\mathcal{T}_i^*$ is a planar graph.

(i) For any face $f \in F(\hat{\mathcal{A}}_i)$, let v_j and v_{j+1} be two consecutive vertices in $\sigma(f)$. Let S be the set of all edges of $\mathcal{T}_i^*$ that intersects v_jv_{j+1}, then $\mathcal{CS}(v_j) \cup \mathcal{CS}(v_{j+1}) = S$.

(ii) As we move from v_j to v_{j+1}, first, we encounter all edges in $\mathcal{CS}(v_j)$ and only then will we encounter the edges of $\mathcal{CS}(v_{j+1})$. Therefore, as we move from v_j to v_{j+1}, the furthest crossing edge of v_j will appear immediately before the furthest crossing edge of v_{j+1}.

Next, consider any vertex $v_j \in \sigma(f)$. We define two points p_{2j} and p_{2j+1} as follows. If v_j does not have any crossing edges, we set $p_{2j} = p_{2j+1} = v_j$. Otherwise, if v_j has a crossing edge, then let e be the furthest crossing edge. We set p_{2j} to be the point of intersection of e with $v_{j-1}v_j$ and set p_{2j+1} to be the point of intersection of v_jv_{j+1} with e. For any j, the following property is true for the segment $\overline{p_{2j}p_{2j+1}}$

- Suppose v_j has a crossing edge. By construction, $\overline{p_{2j}p_{2j+1}}$ is contained in the furthest crossing edge of v_j and the segment $\overline{p_{2j}p_{2j+1}}$ is contained inside the face f.

In addition, the following property is true for $\overline{p_{2j+1}p_{2j+2}}$.

- By construction, p_{2j+1} and p_{2j+2} lie on the edge v_jv_{j+1}. Furthermore, from property (ii), $\overline{p_{2j+1}p_{2j+2}}$ does not intersect with any other edge of $\mathcal{T}_i^*$.

In the following lemma, we formally define the *trace* of a non-triangulated face $f \in F(\hat{\mathcal{A}}_i)$ and non-triangulated face $f' \in F(\mathcal{T}_i^*)$.

Lemma 11. *For any face $f \in \hat{\mathcal{A}}_i$, as we walk along the cycle $C = \overrightarrow{p_0p_1}, \overrightarrow{p_1p_2}, \ldots, \overrightarrow{p_{2m-3}p_0}$, there is a unique non-triangulated face $f' \in F(\mathcal{T}_i^*)$ that appears on the right with respect to the embedding of $\mathcal{T}_i^*$. The face f also appears on the right as we walk along C with respect to the embedding of $\hat{\mathcal{A}}_i$. We define the region enclosed by the cycle C as the* trace *of f, and denote it by $\theta(f) = f \cap f'$ (Figure 4.2).*

Proof. We prove this claim by induction.

Base Case: There are two possibilities. Either (i) $p_0, p_1 = v_0$ or, (ii) p_0 and p_1 are intersection points of the furthest crossing edge e_0 of v_0 with edges $\overrightarrow{v_{m-1}v_0}$ and $\overrightarrow{v_0, v_1}$. In case (i), by construction v_0 is on the boundary of f. We set the face f' to be the face of $\mathcal{T}_i^*$ that lies to the right as we start to walk from v_0 towards v_1 along the edge $\overrightarrow{v_0v_1}$. In case (ii), since e is a crossing edge of v_0, v_0 is convex and $\overline{p_0p_1}$ lies completely inside the face f. Therefore, f will also appear on the right as we walk along this edge. Also, since $\overline{p_0p_1}$ is contained inside e, as we walk from p_0 to p_1, there is a unique face f' that is to the right of e.

Induction Step: Suppose that the faces f and f' appear to the right of all edges starting from vertex p_0 to p_{2j}. We will now show that the claim is also true if we extend the path to p_{2j+1} and then to p_{2j+2}. There are two possibilities: (i) $p_{2j}, p_{2j+1} = v_j$, or (ii) p_{2j} and p_{2j+1} are points of intersection of the furthest crossing edge e_j of v_j.

In case (i), $p_{2j}, p_{2j+1} = v_j$, so the claim is trivially true for the path until p_{2j+1}. $\overline{p_{2j+1}p_{2j+2}}$ will be along the edge v_jv_{j+1}. Note that, as we walk along the path from v_{j-1} through v_j to v_{j+1}, there is no other edge of f' incident on v_j that appears to the right of this path. Otherwise, such an edge will contradict Lemma 7. Therefore, the next line segment $\overline{p_{2j+1}p_{2j+2}}$ will continue to have f' and f on its right.

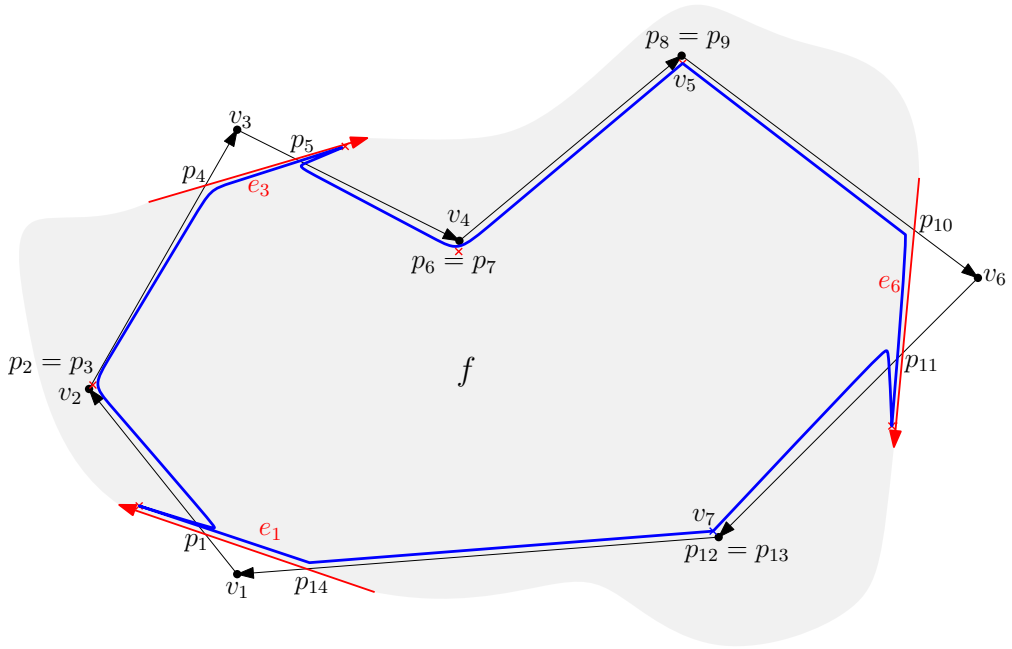

Figure 4.2: The trace, $\theta(f)$, of a non-triangulated face $f \in \hat{\mathcal{A}}_i$. e_1, e_3, and e_6 are the furthest crossing edges for v_1, v_3 and v_6, respectively. Edges added in $\mathcal{W}_i$ are depicted in blue.

In case (ii), p_{2j} and p_{2j+1} are points of intersection of the furthest crossing edge e_j of v_j. Since v_j is a convex vertex, the entire segment $\overline{v_j v_{j+1}}$ is embedded to the right as we walk along from v_{j-1} to v_j. Since p_{2j} is an intersection point of e_j with $v_j v_{j-1}$, and p_{2j+1} is an intersection point of e_j with $v_j v_{j+1}$, we will make a right turn at p_{2j} and p_{2j+1}. When we make a right turn at intersection points, the faces on the right will continue to be on the right side. Therefore, the claim is true for the path until p_{2j+2}. □

The following result also follows from the construction of the trace.

Corollary 12. *For any two distinct faces $f_1, f_2 \in F(\hat{\mathcal{A}}_i)$, $\theta(f_1)$ and $\theta(f_2)$ are disjoint.*

Proof. f_1 and f_2 are disjoint since $\hat{\mathcal{A}}_i$ is planar, therefore, their intersection with any face of $\mathcal{T}_i^*$ will continue to be mutually disjoint. □

Lemma 11 (together with Corollary 12) implies that the trace is a well-defined function mapping a non-triangulated face f in $\hat{\mathcal{A}}_i$ to a single, unique, connected region. Therefore, we can augment $\mathcal{T}_i^*$ to form a new graph, $\widetilde{\mathcal{T}}_i^*$, such that the trace $\theta(f)$ corresponds to a face in $\widetilde{\mathcal{T}}_i^*$, and that $\theta(f)$ is the dominating face for f.

Adding edges to $\mathcal{T}_i^*$ Next, we describe a procedure to embed a set of new edges $\mathcal{W}_i$ to the straight-line embedding of $\mathcal{T}_i^*$ to obtain a new graph $\widetilde{\mathcal{T}}_i^*$ such that $\widetilde{\mathcal{T}}_i^*$ remains a planar

graph. Additionally, for any face f of $\hat{\mathcal{A}}_i$, the trace $\theta(f)$ is contained uniquely inside a face f' of $\widetilde{\mathcal{T}}_i^*$. Furthermore, f' dominates f, so the signature $s(f') \geq s(f)$. Note that the embedding of edges in $\mathcal{W}_i$ that we construct is planar but not necessarily straight-line edges.

$\mathcal{W}_i$ together with a planar embedding of $\mathcal{W}_i$ is constructed as follows. Let $f \in F(\hat{\mathcal{A}}_i)$ have the vertex sequence $\sigma(f) = \langle v_0, \dots v_{m-1} \rangle$. Let the trace of f be given by $\{p_0, \dots, p_{2m-1}\}$ and let e_k be the furthest crossing edge of the convex vertex v_k for each k. Note that the edge e_k exists only if $p_{2k-2} \neq p_{2k-1}$. Recollect that each edge e_k is an edge of a face f' in $\mathcal{T}_i^*$. We direct the edge e_k so that as we walk along e_k the face f' appears on its right. Define ψ, which maps each index $j \in \{0, \dots, m-1\}$ to a vertex v on the boundary of f or f' as follows: if e_j does not exist, then $\psi(j) = v_j$; otherwise, $\psi(j)$ is the vertex that is the head of the directed edge e_j.

Given this map ψ, we now describe a method for adding edges in $\mathcal{W}_i$ that is added to $\widetilde{\mathcal{T}}_i^*$. For every $\overrightarrow{v_j v_{j+1}}$, we add an edge $\overrightarrow{\psi(j)\psi(j+1)} \in \mathcal{W}_i$ and embed this edge to construct a planar embedding of $\widetilde{\mathcal{T}}_i^*$ as follows: Starting at $\psi(j)$, we draw the edge parallel and very close to e_j until it reaches p_{2j-1}. At this point, we will draw the edge parallel and very close to the line segment $\overline{p_{2j-1}p_{2j}}$ until we reach the edge e_{j+1}. Finally, we will continue to draw the edge parallel and very close to the edge e_{j+1} to $\psi(j+1)$. Note that $\overline{p_{2j-1}p_{2j}}$ does not intersect any edge of $\mathcal{T}_i^*$. Note that edges e_j and e_{j+1} are edges of $\mathcal{T}_i^*$, therefore, the newly added edge will not have any intersections with edges of $\mathcal{T}_i^*$. If there are many vertices that have a single edge e^* as a furthest crossing edge, there will be multiple edges that are drawn parallel to the edge e^*. We will draw all of them parallel and close to the edge e^* (carefully stacked, one on top of the other, to avoid intersections). See Figure 4.2 for an example construction of the trace with the newly added edges, and Figure 4.3 for an example where a single edge of $\mathcal{T}_i^*$ is the furthest crossing edges for more than one vertex. This construction ensures that no two edges of $\mathcal{W}_i$ intersect with each other and $\widetilde{\mathcal{T}}_i^*$ is planar. By construction, there is a unique edge added for every $\overrightarrow{v_i v_{i+1}}$. Therefore, there is a unique face $f' \in F(\widetilde{\mathcal{T}}_i^*)$ that contains the trace of any face $f \in F(\hat{\mathcal{A}}_i)$ with $s(f) = s(f')$. From the discussion above, the following lemma follows.

Lemma 13. *The augmented graph $\widetilde{\mathcal{T}}_i^* = \mathcal{T}_i^* \cup \mathcal{W}_i$ can be embedded without intersections. Furthermore, the face $f \in F(\hat{\mathcal{A}}_i)$ has the same signature as the face $f' \in F(\widetilde{\mathcal{T}}_i^*)$ that contains the trace of f.*

However, we may create multiple copies of the same edges in $\mathcal{W}_i \cup \mathcal{T}_i^*$ (see Figure 4.3). To overcome this difficulty, we remove duplicate copies of the same edge. Removal of an edge from a planar graph will merge two faces h and h' into a single face h'' where h'' has a larger signature than either h or h'. In Lemma 14, we show that no edge with multiple copies participates in more than one dominating face. Therefore, removal of an edge only increases the signature of the dominating face it participates in, therefore, the face remains a dominating face.

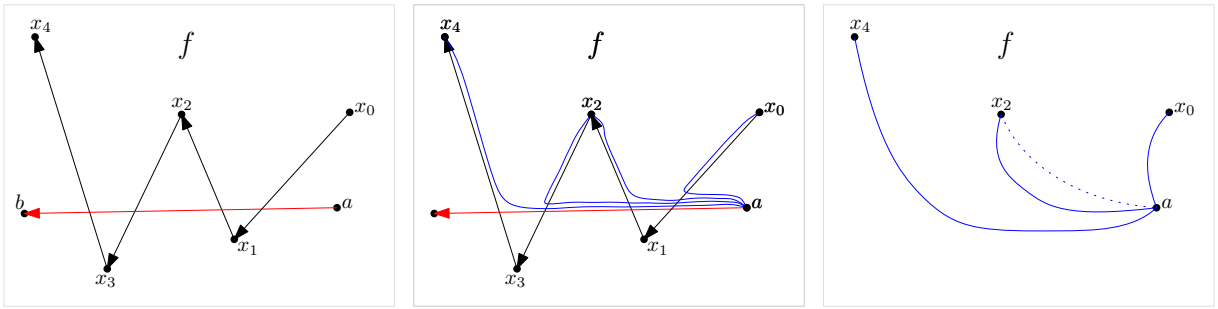

Figure 4.3: Duplicate edges created in the construction of $\mathcal{W}_i$. (a) the edge $ab \in \mathcal{T}_i^*$ intersects the face f in $\hat{\mathcal{A}}_i$. The boundary vertex sequence of f at this point is $\langle x_0, x_1, x_2, x_3, x_4 \rangle$ which contributes 4 to $s(f)$. (b) Edges in $\mathcal{W}_i$ are added and embedded as depicted in blue. (c) The edges of $\mathcal{W}_i$. Note that there are multiple edges between a and x_2, we show that we can delete one copy. This results on a potion on the boundary which also contributes 4 to the signature.

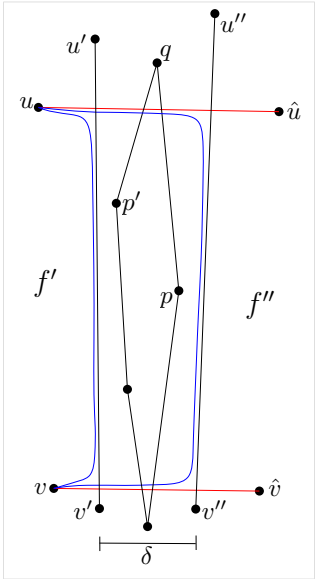

Figure 4.4: There can be no trace between two duplicate edges in $\mathcal{W}_i$ as shown in Lemma 14.

Lemma 14. *For any edge uv that has two or more copies in $\widetilde{\mathcal{T}}_i^*$ between u and v, no copy of uv can participate in two distinct dominating faces.*

Proof. Let there be two edges uv embedded in $\widetilde{\mathcal{T}}_i^*$. We assume both these edges were added in $\mathcal{W}_i$. The case where only one edge was added in $\mathcal{W}_i$ is easier. We assume that the duplicate edge was added due to an edge $u'v'$ and $u''v''$ of $\hat{\mathcal{A}}_i$ that intersected faces $f', f'' \in F(\mathcal{T}_i^*)$, because and edge was added to $\mathcal{W}_i$ for both $u'v'$ and $u''v''$. Note that $u'v'$ and $u''v''$ appear on the boundary of f' and f'', and that these edges must intersect at least one edge of f'. We present our proof for the case where $u'v'$ and $u''v''$ intersect two edges $e_u = u\hat{u}$ and $e_v = v\hat{v}$ on the boundary of f'. See Figure 4.4 for the construction. For the sake of contradiction, let there be another non-triangulated face $\tilde{f} \in \hat{\mathcal{A}}_i$ whose trace is in the region between the two embeddings of uv. In this case, there must be at least two vertices of $\tilde{f}$ that lie inside this region (otherwise $\tilde{f}$ will either be a triangle or $\theta(f')$ will not intersect this region). At least one of these points p' will have a non-adjacent edge pq of $\tilde{f}$ that is visible to it. Since $u\hat{u}$ and $v\hat{v}$ have a length at most $\delta = \frac{\gamma 3^{i-1}}{\sqrt{2}}$, this immediately implies pq is δ-visible to p'. In addition, it follows that both edges incident on p' are δ-visible to pq, since a point arbitrarily close to p' on either of the edges incident to p' (and on the boundary of $\tilde{f}$) must also be δ-visible. If both of these edges are adjacent to pq, then $\tilde{f}$ was not a non-triangulated face. If either of these edges are not adjacent to pq, it contradicts Lemma 7, since edges in $\hat{\mathcal{A}}_i$ correspond to maximal triangulated chains. A similar argument extends to all other cases. □

It follows that $\widetilde{\mathcal{T}}_i^*$ is a planar graph that dominates $\hat{\mathcal{A}}_i$. Thus, by Corollary 10, Invariant 2 follows. Since both Invariant 1 (after proving properties (P1)–(P2) in Section 4.5) and Invariant 2 holds, this completes the proof of Theorem 1.

4.5 Proving Properties of a Maximal PSLG in $\mathcal{G}_i$

In the following section we prove (P1)–(P4) presented in Section 4.1, which were used in the proof of Invariant 1. First recall that by Lemma 2, we know that vertices in 1-chains are at most 3 cells apart, and vertices participating in a 2-chain are at most 4 cells apart. This restricts the number of possible configurations of cells in which vertices participating in either type of chain can appear. We can deduce certain properties that arise due to the possible configurations, expressed in Lemma 15 and Lemma 16, which will be used in the proof of the properties.

Lemma 15. *Let C_1, C_2, and C_3 be three cells in G_i such that $C_1 \in N(C_2)$ and $C_2 \in N(C_3)$ (note C_1 and C_3 are 3 cells apart). Let H be the region enclosed by the convex hull of C_1, C_2 and C_3. Then,*

(a) if $C_3 \in N(C_1)$, then $H \subset \mathcal{N}(C_1) \cap \mathcal{N}(C_2) \cap \mathcal{N}(C_3)$,

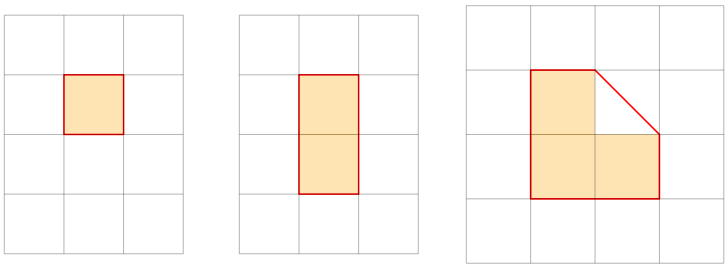

Figure 4.5: Three mutually adjacent cells.

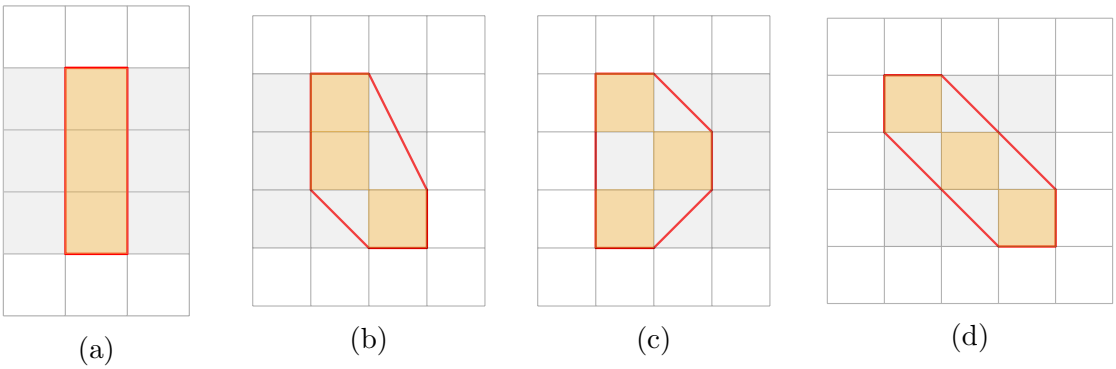

Figure 4.6: Possible configurations for adjacent cells that are three cells apart.

(b) otherwise $H \subset (\mathcal{N}(C_1) \cap \mathcal{N}(C_2)) \cup (\mathcal{N}(C_2) \cap \mathcal{N}(C_3))$.

In particular, in either case $H \subset \mathcal{N}(C_2)$.

Proof. For both (a) and (b) it is not difficult to establish the result by considering all possible configurations of three cells which satisfy the necessary conditions. Let C_1, C_2, and C_3 be as above.

Suppose (a) $C_3 \in N(C_1)$, so these three cells are mutually adjacent. Not counting symmetry, there are only three possible cell configurations that allow three cells to be mutually adjacent: either the three cells are equal (Figure 4.5a), or exactly two of the cells are equal, and the third is adjacent (Figure 4.5b), or all three cells are distinct (Figure 4.5c). In all three cases, the convex hull H of the three cells is contained in the neighborhood of each cell, i.e. $H \subset \mathcal{N}(C_1) \cap \mathcal{N}(C_2) \cap \mathcal{N}(C_3)$.

Now suppose (b) $C_3 \notin N(C_1)$. Then there are exactly four possible configurations (not counting symmetry) to arrange the cells, each of which are depicted in Figure 4.6. In each of these cases it is not difficult to see that any point in H must be contained in the neighborhood of at least two cells: $H \subset (\mathcal{N}(C_1) \cap \mathcal{N}(C_2)) \cup (\mathcal{N}(C_2) \cap \mathcal{N}(C_3))$. Both (a) and (b) immediately implies $H \subset \mathcal{N}(C_2)$. □

Lemma 16. *Let C_1, C_2, C_3, and C_4 be four cells in G_i such that $C_1 \in N(C_2)$, $C_2 \in N(C_3)$, $C_3 \in N(C_4)$ (so C_1 and C_4 are 4 cells apart). Let H be the region enclosed by the convex hull of C_1, C_2, C_3, and C_4. Then $H \subset (\mathcal{N}(C_2) \cup \mathcal{N}(C_3))$ in all but two cases.*

Proof. We establish this claim by considering all possible cell configurations. If the four cells are not distinct, this claim is reduced to Lemma 15. Therefore, consider only the twelve possible configurations (not counting symmetry) of four distinct cells satisfying these conditions depicted in Figure 4.7. Except for the two depicted in Figure 4.7j and Figure 4.7k, the claim holds for all other cases. □

We now describe a "sweeping" procedure which will be re-used in subsequent proofs of the properties in order to show that a particular region is void of input points of P other than those on the boundary of the region. Fix a vertex u, and an edge xy. We *sweep* $\overline{vx}$, starting at x, along xy towards y if we define $\alpha(t) = x(1-t) + yt$ for $t \in [0,1]$ and we consider the line segment $\overline{v\alpha(t)}$ for t ranging from 0 to 1. We call $\overline{v\alpha(t)}$ the *sweep line*. Therefore, one can imagine the line starting at $\overline{vx}$ for $t = 0$ sweeping across a face by sliding x along $\overline{xy}$, and reaching $\overline{vy}$ at time $t = 1$. The starting and ending points need not be endpoints of an edge. If $p, q \in \overline{xy} \cup \{x, y\}$ is the starting and ending point, respectively, then one simply defines $\alpha(t)$ to be the appropriate parameterization of $\overline{pq} \subset \overline{xy}$. In this case, we say that, starting at q, we sweep $\overline{vp}$ along xy towards q.

We now proceed to prove (P1) through (P4). Let $\mathcal{A}$ be a maximal PSLG with respect to $\mathcal{G}_i$. Let f be a non-triangulated face in $\mathcal{A}$ with the boundary vertex sequence $\sigma(f)$.

(P1) Suppose $v_j \in \sigma(f)$ is convex. Then v_{j-1}, v_j, and v_{j+1} are in three distinct cells $C_{v_{j-1}}$, C_{v_j}, and $C_{v_{j+1}}$ of G_i, respectively, and $C_{v_{j-1}} \notin N(C_{v_{j+1}})$.

Proof. The fact that v_{j-1}, v_j, and v_{j+1} appear consecutively in $\sigma(f)$ implies that the edges $v_{j-1}v_j$ and v_jv_{j+1} are in $\mathcal{A}$, hence $v_j \in \mathcal{N}(v_{j-1})$ and $v_j \in \mathcal{N}(v_{j+1})$. Suppose, for the sake of contradiction, that $v_{j-1} \in \mathcal{N}(v_{j+1})$, then v_j, v_{j-1}, and v_{j+1} are in mutually neighboring cells which satisfy the conditions of Lemma 4.5(a). So H, the region enclosed by the convex hull of C_{v_j-1}, C_{v_j}, and C_{j+k}, is contained in $\mathcal{N}(v_{j-1}) \cap \mathcal{N}(v_j) \cap \mathcal{N}(v_{j+1})$. Since any point in the triangle formed by v_{j-1}, v_j, and v_{j+1} must be contained in H, it follows that such a point is also contained in $\mathcal{N}(v_{j-1}) \cap \mathcal{N}(v_j) \cap \mathcal{N}(v_{j+1})$. Sweep the line segment $\overline{v_{j-1}v_j}$ along v_jv_{j+1} towards v_{j+1}. Let t^* be the smallest t such that the sweep line $\overline{v_{j-1}\alpha(t^*)}$ intersects a vertex v^*. Then v^* is visible to both v_{j-1} and v_j, otherwise v^* could not have been the first vertex $\overline{v_{j-1}\alpha(t)}$ intersects. Additionally, since $v^* \in H$, v^* must be in $\mathcal{N}(v_j)$ and $\mathcal{N}(v_{j-1})$. Since $\mathcal{A}$ is maximal, it follows that both the edges $v_{j-1}v^*$ and v_jv^* will be present in $\mathcal{A}$, hence v_{j-1} and v_j do not appear directly after one another in $\sigma(f)$, contrary to assumption, so this case is impossible. If no such v^* is encountered, then it implies that v_{j+1} is visible to v_{j-1}, hence the edge $v_{j-1}v_{j+1}$ is present in $\mathcal{A}$, and v_{j-1}, v_j, and v_{j+1} forms a triangulated face, contradicting the assumption that v_{j-1}, v_j, and v_{j+1} appears in this order in $\sigma(f)$. Therefore, one may

Figure 4.7: Possible configurations for adjacent cells that are four cells apart.

conclude that $v_{j-1} \notin \mathcal{N}(v_{j+1})$, and hence v_{j-1}, v_j, and v_{j+1} appears in three distinct cells, as claimed. □

(P2) Suppose $v_j \in \sigma(f)$ and $v_k v_{k+1}$ is any edge on the boundary of f such that $v_k \in \mathcal{N}(v_j)$ (resp. $v_{k+1} \in \mathcal{N}(v_j)$) and $v_k v_{k+1}$ is visible to v_j in $\sigma(f)$. Then,

(i) the chain $\mathbb{C}$ from v_j to v_{k+1} (resp. chain $\tilde{\mathbb{C}}$ from v_k to v_j) in $\sigma(f)$ is a 1-chain, and,

(ii) v_{k+1} is a forward (resp. v_k is backward) support vertex for every vertex from v_j to v_{k-1} (resp. from v_{k+2} to v_j).

Proof. We begin by showing (i), and (ii) follows. Suppose $v_k \in \mathcal{N}(v_j)$. Let T be the triangle enclosed by the convex hull of the three points v_j, v_k, and v_{k+1}, which is contained in the convex hull of the three cells containing v_j, v_k, and v_{k+1}. Note that these three vertices satisfy the conditions of Lemma 15, therefore, $T \subset ((\mathcal{N}(v_j) \cap \mathcal{N}(v_k)) \cup (\mathcal{N}(v_k) \cap \mathcal{N}(v_{k+1}))$. Since $v_k v_{k+1}$ is visible to v_j, there exists a point $q \in v_k v_{k+1}$ such that q is visible to v_j, and the line segment $\overline{v_j q}$ does not intersect any edges of $\mathcal{A}$ and divides the triangle T into two triangles, T_1 and T_2. Let T_1 be the triangle formed by v_j, v_k, and q, and let T_2 be the triangle formed by v_j, q, and v_{k+1}. We show that all input points in T must be boundary vertices of f which appear in $\sigma(f)$.

First we show that the triangle T_2 does not contain any input points in its interior. Starting at q, sweep the line $\overline{v_j q}$ along $v_k v_{k+1}$ towards v_{k+1}. Let t^* the smallest t such that the sweep line $\overline{v_j \alpha(t^*)}$ intersects an input point u^*; u^* is contained in $T_2 \subset \mathcal{N}(v_k)$. Then either v_k is visible to u^*, or there is an edge blocking v_k from u^* but an endpoint u of such an edge must be contained in T_1 (otherwise the sweep line would have encountered it), and $u \in \mathcal{N}(v_k)$. Then either edge $v_k u^*$ or $u u^* \in \mathcal{A}$ since $\mathcal{A}$ is maximal, however, such an edge would intersect $\overline{v_j q}$, a contradiction. Therefore, the sweep line reaches $t = 1$ without intersecting points, which implies that T_2 does not contain any input points in its interior, and therefore v_j is visible to v_{k+1} in $\sigma(f)$. It follows that if all vertices from v_{j+1} to v_{k-1} is reflex, all these vertices would be visible to v_{k+1}, which would allow one to conclude $\mathbb{C}$ is a 1-chain.

Now we show that any input point in T_1 is in σ and vertices between v_j and v_k must be reflex. Also, v_k is the only convex vertex between v_j and v_{k+1} in $\sigma(f)$. Starting at q sweep $\overline{v_j q}$ along $v_k v_{k+1}$ towards v_k. Let t' be the smallest t such that the sweep line $\overline{v_j \alpha(t')}$ intersects a vertex w_0. If there is no such w_0, then v_k is visible to v_j and we are done, since v_k is convex and there are no other vertices in $\mathbb{C}$. So $\mathbb{C}$ is a 1-chain. If there is such a w_0, it follows that w_0 is visible to v_j. By Lemma 15 $w_0 \in (\mathcal{N}(v_j) \cap \mathcal{N}(v_k)) \cup (\mathcal{N}(v_k) \cap \mathcal{N}(v_{k+1}))$. If $w_0 \in \mathcal{N}(v_k) \cap \mathcal{N}(v_{k+1})$, then w_0 is visible to v_{k+1} by the prior two sweeping arguments, then the edge $w_0 v_{k+1} \in \mathcal{A}$, a contradiction since $w_0 v_{k+1}$ intersects $\overline{v_j q}$. Thus, $w_0 \in \mathcal{N}(v_j) \cap \mathcal{N}(v_k)$, in particular, $w_0 \in \mathcal{N}(v_k)$. Since $\mathcal{A}$ is maximal, $v_j w_0 \in \mathcal{A}$, so w_0 is the first vertex after v_j in $\sigma(f)$ and w_0 is reflex (otherwise w_0 is not the first vertex the sweeping line intersects). By the prior sweeping arguments, the triangle formed by w_0, $\alpha(t')$, and v_{k+1} does not contain

an input point, and therefore v_{k+1} is visible to w_0. Repeat the same argument, by sweeping the line $\overline{w_0\alpha(t')}$ along v_kv_{k+1} towards v_k, starting at $\alpha(t')$. If you encounter a vertex, the first vertex w_1 you encounter will be reflex, $w_0w_1 \in \mathcal{A}$, $w_1 \in \mathcal{N}(v_k)$, and v_{k+1} is visible to w_0. Otherwise, v_k is visible to w_0 and you are done. Repeat this procedure until you reach w_r such that w_r is visible to v_k. Then $w_0, \ldots, w_r$, are all reflex and contained in $\mathcal{N}(v_k)$. It follows that v_k is the only convex vertex, and so $\mathbb{C}$ is a 1-chain, which shows (i). It now follows immediately that v_k is the forward convex vertex for every vertex from v_j to v_{k-1}, and therefore by definition, v_{k+1} is the forward support vertex of these vertices. A symmetric argument holds if we assume $v_{k+1} \in \mathcal{N}(v_j)$ (in which case v_k will be the backwards support vertex of all vertices between v_{k+1} and v_j in $\sigma(f)$). □

(P3) For any chain $\mathbb{C}(v, y)$ from v to y in $\sigma(f)$,

(i) if $\mathbb{C}(v, y)$ is a 1-chain with v' as its only convex vertex, then the region $E = E(v, y)$ is contained in $\mathcal{N}(v')$, i.e., $E \subset \mathcal{N}(v')$, and E contains no input points of P, other than those in $\mathbb{C}(v, y)$.

(ii) if the chain $\mathbb{C}(v, y)$ is a 2-chain with v' and v'' as the two convex vertices, then the region $E = E(v, y)$ is such that $(E \cap (\mathcal{N}(v') \cup \mathcal{N}(v'')) \cap P$ contains only points of C. In only two cases, $E \not\subset (\mathcal{N}(v') \cup \mathcal{N}(v''))$ (see Figure 4.1 and Figures 4.7j and 4.7k). In all other cases, E contains only points of C, but no other points of P.

Proof. We first show (i). Suppose $\mathbb{C}(v, y) = \mathbb{C}$ is a 1-chain with v' as its only convex vertex. Let $E = E(v, y)$. Since $\mathbb{C}$ is a 1-chain, it follows that v is visible to y. Let H be the region enclosed by the convex hull of the three cells containing v, v', and y. By definition of a 1-chain, these three satisfy the conditions for Lemma 15, and hence $H \subset \mathcal{N}(v_k)$. Since $E \subset H$, it follows that $E \subset \mathcal{N}(v_k)$. Starting at v, sweep the line segment $\overline{v'v}$ along $\overline{vy}$ towards y. Note that v' is not necessarily visible to v, since there may be reflex vertices in the chain from v to v', so the edge $v'v$ may not be present. If the edge is present, then v' appears immediately after v in $\sigma(f)$. Therefore, as the sweep line $\overline{v'\alpha(t)}$ is being swept towards y, it might intersect a vertex on the chain $\mathbb{C}$, a boundary vertex of f. Let t^* be the smallest t for which the sweep line intersects an input point v^* that is not in $\mathbb{C}$. Then this point is in E, and hence $v^* \in \mathcal{N}(v')$. Additionally, v^* is visible to v', since the only vertices that are present in the region that has been swept by the sweeping line must be reflex, therefore, the edge $v'v^* \in \mathcal{A}$. However, this implies that v^* must be in the chain $\mathbb{C}$, a contradiction. Therefore, the only points of P contained in E are those that are also in $\mathbb{C}$.

To show (ii), suppose $\mathbb{C}$ is a 2-chain such that v' and v'' are its two convex vertices. Let H be the region enclosed by the convex hull of the four cells containing v, v', v'', and y. Then these four cells satisfy the conditions for Lemma 16, so for all but two cases, $H \subset (\mathcal{N}(v') \cup \mathcal{N}(v''))$. Then $E \subset H \subset (\mathcal{N}(v') \cup \mathcal{N}(v''))$, except for two cases. Other than those two exceptions, the same argument as in (i) holds to show that any input point in E must be a vertex also in $\mathbb{C}$. Hence, one may conclude the result. □

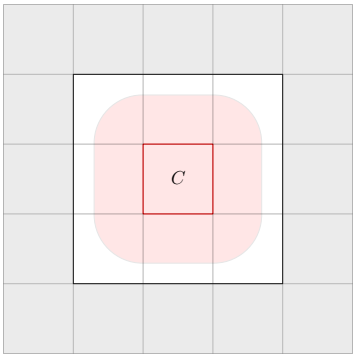

Figure 4.8: The set J (depicted by the red shaeded region), for the cell C.

(P4) For any vertex $v \in \sigma(f)$, if an edge xy on the boundary of f is δ-visible for $\delta = \frac{3^{i-1}}{\sqrt{2}}$, then exactly one of x and y are in $\mathcal{N}(v)$ and the other is not.

Proof. Let C be the cell in grid G_i that contains v. Then let $J = \bigcup_{p \in C}\{z \in \mathbb{R}^2 : \|pz\| \leq \delta\}$ (see the red region in Figure 4.8). Since xy is δ-visible to v, there exists a point $q \in \overline{xy}$, such that q is visible to v, and $q \in J$. Suppose, for the sake of contradiction, that (a) $x, y \in \mathcal{N}(v)$ or (b) $x, y \notin \mathcal{N}(v)$. We consider the cases separately.

(a) $x, y \in \mathcal{N}(v)$. Without loss of generality, assume v, x, and y appear in this order in $\sigma(f)$ (a symmetric argument holds if the order is y, x, v). Since q is visible to v in $\sigma(f)$, the line segment $\overline{vq}$ does not intersect any edges of $\mathcal{A}$. Starting at q, sweep $\overline{v\alpha(t)}$ along xy towards x. Let t^* be the smallest t such that the sweep line $\overline{v\alpha(t^*)}$ intersects an input point v^*; if the sweep line intersects no input point, let $v^* = x$. Then v^* is visible to v so that $vv^* \in \mathcal{A}_i$. Similarly, sweep $\overline{vq}$ towards y, starting at q and moving from q along xy towards y. Let t' be the smallest t such that this sweep line intersects a vertex v', or if the sweep line intersects no vertex, set $v' = y$. Then v and v' are visible to each other, hence $vv' \in \mathcal{A}$. In addition, v' and v^* are in each other's neighborhoods so that $v'v^* \in \mathcal{A}$. However, this implies v, v', and v^* forms a triangulated face and $\overline{vq}$ is not contained in f, contradicting the fact that q is visible to v. Therefore, this case is impossible.

(b) $x, y \notin \mathcal{N}(v)$. Note that since $xy \in \mathcal{A} \subset \mathcal{G}_i$, x and y must be in neighboring cells. Furthermore, the cells containing x and y must be adjacent to $N(v)$ (the grey region in Figure 4.8). It is not difficult to see that an edge between x in y in two grey cells in Figure 4.8 cannot intersect J. The result follows. □

Chapter 5

Concluding Remarks

We introduced a polynomial time approximation algorithm for the MWT with a worst-case approximation ratio of 24, and an expected approximation ratio of 16. This is achieved by partitioning edges into levels using grids, and applying a variant of the ring heuristic at each level i to obtain a partial candidate solution that we show has a cardinality greater than a restricted optimal solution at level $i-1$.

Improving the Algorithm. The approximation ratio can be improved to 21 (and the expected approximation ratio to 14) by making the following simple modification to our algorithm (cf. Section 3.1) based on the following observation. For any triangulated chain $\mathbb{C}(v_{j+k}, v_p)$ generated by the algorithm, $\mathbb{C}(v_{j+k}, v_p)$ is a 2-chain only if the algorithm executes step 1(a) or 1(c). Using the same notation as in the presentation of the algorithm, consider the case where $\mathbb{C}(v_{j+k}, v_p)$ is generated in 1(a) (so $v_p = v_s$). We have shown in Lemma 3 that edges in $\mathcal{E}(v_{j+k}, v_s)$ that triangulate $E(v_{j+k}, v_s)$ are bounded by $4\sqrt{2} \cdot \gamma 3^{i-1}$. However, by construction $\mathbb{C}(v_{j+k}, v_l)$ and $\mathbb{C}(v_{l+1}, v_s)$ are both 1-chains, so if both chains could connect to their support vertices, edges in $\mathcal{E}(v_{j+k}, v_s)$ would be bounded by $3\sqrt{2} \cdot \gamma 3^{i-1}$. Since edges added for one chain hide the support vertex of the other, both sets of edges cannot be added without violating planarity, so there is a conflict. The algorithm resolves this conflict by always connecting edges for $\mathbb{C}(v_{j+k}, v_{l-2})$ to its forward support and adding longer edges (with endpoints 4 cells apart) by adding edges from $\mathbb{C}(v_{l+1}, v_s)$ to v_{j+k}. Equivalently, one can add the longer edges from $\mathbb{C}(v_{j+k}, v_{l-2})$ to v_s and add edges from $\mathbb{C}(v_{l+1}, v_s)$ to its backward support v_{j+k}. By adding a check in the algorithm to make the choice which results in the smaller weight for edges in $\mathcal{E}(v_{j+k}, v_s)$, we can make the argument that the upper bound on the average length of an edge added to $\hat{\mathcal{A}}_i$ is $\frac{7}{2}\sqrt{2} \cdot \gamma 3^{i-1}$. This helps improve the ratio to 21 (or expected 14).

Lemma 17. *With a slight modification to step 1(b) of the algorithm, we can guarantee that no more than half the edges in $\hat{\mathcal{A}}_i \setminus \mathcal{A}$ are 4 cells apart and at least half the edges are no more than 3 cells apart. In addition, both invariants still hold.*

Proof. Using the notation used in the description of the algorithm (cf. Chapter 3), this modification is to step 1(b) of the algorithm:

1 (b) If $\mathbb{C}(v_{j+k}) = \mathbb{C}(v_{j+k}, v_l)$, then for convenience set $\mathbb{C}(v_{l+1}, v_s) \leftarrow \emptyset$ (a chain with no vertices. Then,

- if $|\mathbb{C}(v_{j+k}, v_l)| \geq |\mathbb{C}(v_{l+1}, v_s)|$, add an edge from v_l to every vertex in $\mathbb{C}(v_{j+k}, v_{l-2})$ and add edges from v_{j+k} to every vertex in $\mathbb{C}(v_{l+1}, v_s)$;
- if $|\mathbb{C}(v_{j+k}, v_l)| < |\mathbb{C}(v_{l+1}, v_s)|$, add edges from v_{l-1} to every vertex in $\mathbb{C}(v_{l+1}, v_s)$ and add an edge from v_s to every vertex in $\mathbb{C}(v_{j+k}, v_{l-2})$.

Given this modification it is not difficult to see that at most half the edges added in Phase 1 of the algorithm are 4 cells apart, and the remainder are at most 3 cells apart. It follows that for any edge $uv \in \hat{\mathcal{A}}_i \setminus \mathcal{A}_i$, the average length $\|uv\| \leq \frac{7}{2}$. In addition, the same arguments used to prove that planarity is maintained when adding edges in Phase 1 for Invariant 1 remain valid. Furthermore, it is not difficult to see that cardinality arguments used in the proof of Invariant 2 are unaffected by this modification. □

Corollary 18. *Suppose that Lemma 17 holds in addition to Invariants 1 and 2. Then the candidate solution produced by the algorithm with the modification described in Lemma 17 is an α-approximalte MWT where $\alpha \leq 21$. Furthermore, the expected value of $\alpha \leq 14$.*

Proof. Since the argument in the proof of Theorem 1 remains valid for an average weight (since we are simply considering the ratio between the sum of edge weights), by the same argument the approximation ratio improves to 21 and the expected approximation goes down to 14. □

Bottleneck Triangulation and P-Norm. In addition to our algorithm's minimizing the sum of edge weights, it minimizes the weight of the edge with the maximum weight. One may define $w(\mathcal{T})$ to be the maximum weight of any edge in the triangultions, which is known as the bottleneck distance. Therefore, our algorithm also produces a constant approximation for the MWT, with weight redifined in this way. This is clear from our analysis, since every edge in our candidate solution is within the same constant of at least one edge in the optimal MWT. Another possible definition of weight could be given by the p-norm, therefore, $w(\mathcal{T}) = \left(\sum_{ab \in \mathcal{T}} \|ab\|^p\right)^{1/p}$. In this case, our analysis remains valid since we compare ratios of edges weights, and for this weight function it is sufficient to compare $\|ab\|^p$ for pairs of edges, and therefore our algorithm will still compute a constant approximation of the MWT. These guarantees are not clear from the analysis given by Plaisted and Hong [14] for their ring heuristic. Their analysis relies on the triangle inequality, which does not apply for $\|ab\|^p$ for $p > 1$, so analysis cannot be extended in the same manner. Both these additional guarantees imply that the triangulation produced by our algorithm ensures some measure of regularity, which is important for many applications.

A PTAS for the MWT? We are optimistic that the techniques used here can be adapted in order to develop a PTAS for the MWT. The construction would be maintaining multiple candidate solutions at each grid level, and using dynamic programming to make near optimal choices at each step, where optimal in our case would be reflected in the candidate that is closest (in the topological sense) to the restricted optimal triangulation. A difficulty in extending this approach in this manner is that different PSLGs can have the same cardinality but with different topologies. It is possible that one candidate solution leads to the optimal, and the other deviates from the optimal in such a way that making the guarantees about the relationship between the cardinality of the candidate and the optimal solutions becomes challenging.

Bibliography

[1] S. Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, 45(5):753–782, 1998.

[2] M. W. Bern and D. Eppstein. Mesh generation and optimal triangulation. In Ding-Zhu Du and Frank Kwang-Ming Hwang, editors, *Computing in Euclidean Geometry*, number 1 in Lecture Notes Series on Computing, pages 23–90. World Scientific, 1992.

[3] K. L. Clarkson. Approximation algorithms for planar traveling salesman tours and minimum-length triangulations. In *Proceedings of the Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '91, pages 17–23, Philadelphia, PA, USA, 1991. Society for Industrial and Applied Mathematics.

[4] G. B. Dantzig, A. J. Hoffman, and T. C. Hu. Triangulations (tilings) and certain block triangular matrices. *Mathematical Programming*, 31(1):1–14, 1985.

[5] R. D. Düppe and H. J. Gottschalk. Automatische interpolation von isolinien bei willkürlich verteilten stützpunkten. *Allgemeine Vermessungs-Nachrichten*, 77:423–426, 1970.

[6] D. Eppstein. Approximating the minimum weight steiner triangulation. *Discrete Comput. Geom.*, 11(2):163–191, 1994.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP–Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[8] D. S. Hochbaum. Various notions of approximations: Good, better, best, and more. In D. S. Hochbaum, editor, *Approximation algorithms for NP-hard problems*, chapter 9, pages 346–398. PWS Pub. Co, Boston, 1997.

[9] C. Levcopoulos and D. Krznaric. Quasi-greedy triangulations approximating the minimum weight triangulation. *J. Algorithms*, 27(2):303–338, May 1998.

[10] E. L. Lloyd. On triangulations of a set of points in the plane. SFCS '77, pages 228–240, Washington, DC, USA, 1977. IEEE Computer Society.

[11] G. K. Manacher and A. L. Zobrist. Neither the greedy nor the Delaunay triangulation of a planar point set approximates the optimal triangulation. *Inform. Process. Lett.*, 9(1):31–34, 1979.

[12] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple new method for the geometric k-mst problem. In *Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '96, pages 402–408, Philadelphia, PA, USA, 1996. Society for Industrial and Applied Mathematics.

[13] W. Mulzer and G. Rote. Minimum-weight triangulation is NP-hard. *CoRR*, abs/cs/0601002, 2006.

[14] D. A. Plaisted and J. R. Hong. A heuristic triangulation algorithm. *J. Algorithms*, 8(3):405–437, 1987.

[15] J. Remy and A. Steger. A quasi-polynomial time approximation scheme for minimum weight triangulation. *J. ACM*, 56(3):15:1–15:47, May 2009.

[16] M. I. Shamos and D. Hoey. Closest-point problems. In *Proceedings of the 16th Annual Symposium on Foundations of Computer Science*, pages 151–162, Washington, DC, USA, 1975. IEEE Computer Society.

[17] A. Yousefi and N. E. Young. On a linear program for minimum-weight triangulation. *CoRR*, abs/1111.5305, 2011.