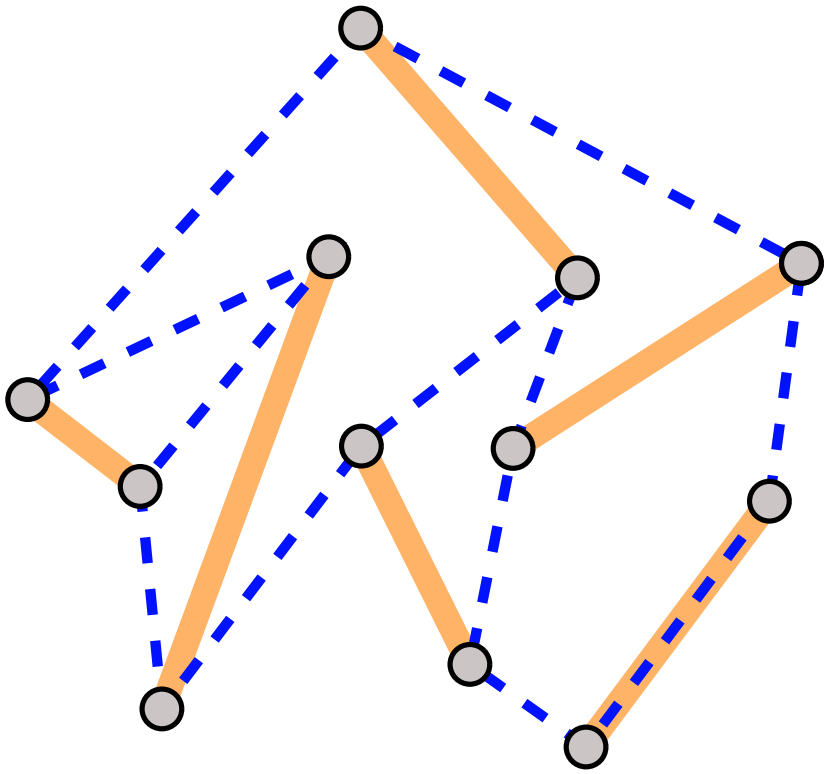
Diss. ETH No. 15897

Michael Hoffmann

On the Existence of Paths and Cycles



2005

DISS. ETH No. 15897, 2005

On the Existence of Paths and Cycles

A dissertation submitted to the
Swiss Federal Institute of Technology Zürich
for the degree of Doctor of Technical Sciences

presented by
Michael Hoffmann
Dipl. Math., Freie Universität Berlin, Germany
born May 6, 1970 in Berlin, Germany
citizen of Germany

accepted on the recommendation of
Prof. Emo Welzl, ETH Zürich, examiner
Prof. Erik Demaine, MIT, Cambridge, co-examiner

Abstract

This thesis investigates questions related to the existence of certain paths and cycles in graphs. Its major part is centered around the following question: Given a set of n line segments in the plane, can one connect all segment endpoints by a closed path that does not cross itself nor any of the segments? In other words, under which conditions can one find a simple polygon P whose vertices are the segment endpoints and such that P does not cross any segment? (The segments may appear as edges of P.)

Such polygons are known as *Hamiltonian polygons* and the motivation to study them is twofold:

- Traversals of line segments are a natural generalization of the Euclidean Traveling Salesman Problem (ETSP) for points in the plane.
- Statements about the existence of paths or cycles through line segments are structural results about the so-called visibility graph of the segments. Visibility graphs are important geometric structures that appear frequently in applications, for example in rendering and in the context of shortest path queries.

As a main result we show that a Hamiltonian polygon exists for every finite set of disjoint line segments that are not all collinear.

The existence of a Hamiltonian polygon is interesting as a structural result that can be used to analyze properties of other combinatorial structures. For example, we can use it to obtain asymptotically tight bounds on the size of so-called *alternating paths*: in every set of n disjoint line segments there are $\Omega(\log n)$ segments that can be connected to form a simple path on their endpoints (the path passes through all these segments from one endpoint to the other) which does not intersect any of the remaining segments.

Furthermore, we consider a question that has arisen in the context of service deployment in communication networks: finding *chordless paths* in graphs. A chordless path is a path for which no two nonconsecutive vertices are connected by an edge in the graph. The problem CP3V is to decide for three given vertices u, v, w of a graph whether they can be connected by a chordless path from u via v to w that consists of at most k vertices.

Here our result is negative: We show that CP3V is $W[1]$-hard, that is, it is unlikely that the problem can be solved in time polynomial in the size of the input graph, even if an arbitrary, for example, exponential dependence on k is permitted. The reduction extends to a number of related problems for directed graphs. In particular, deciding on the existence of a chordless (u, w)-path on at most k vertices in a directed graph is $W[1]$-complete with respect to k.

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Existenz von bestimmten Pfaden und Kreisen in Graphen. Ein Hauptaugenmerk liegt hierbei auf folgendem geometrischen Problem: Betrachte n Liniensegmente in der Ebene; kann man deren Endpunkte durch einen geschlossenen Polygonzug verbinden, welcher weder sich selbst noch eines der Segmente kreuzt? Anders ausgedrückt: Unter welchen Voraussetzungen gibt es ein einfaches Polygon, dessen Eckpunkte genau die Endpunkte der Liniensegmente sind, und dessen Kanten keines der Segmente kreuzen? (Die Segmente dürfen aber als Kanten des Polygons auftreten.)

Derartige Polygone sind als *Hamiltonsche Polygone* bekannt, und es gibt eine Reihe von Gründen, sich mit ihnen zu befassen:

- Polygonzüge durch Liniensegmente sind eine natürliche Verallgemeinerung des Rundreise-Problems für Punkte in der Euklidischen Ebene,
- und Aussagen über die Existenz von Pfaden und Kreisen durch Liniensegmente sind auch strukturelle Aussagen über den sogenannten Sichtbarkeitsgraphen der Segmente. Sichbarkeitsgraphen sind geometrische Strukturen von fundamentaler Bedeutung; sie tauchen in vielen Bereichen auf, zum Beispiel in der Computer-Graphik oder im Zusammenhang mit der Berechnung von kürzesten Wegen in einer Umgebung mit Hindernissen.

Ein Hauptergebnis dieser Arbeit ist, dass für jede endliche Menge von Liniensegmenten in der Ebene ein Hamiltonsches Polygon existiert, sofern nicht alle Segmente auf einer gemeinsamen Geraden liegen.

Mit Hilfe dieses Theorems können wir auch asymptotisch scharfe Schranken für die Grösse sogenannter *alternierender Pfade* beweisen: Für jede Menge von n Liniensegmenten in der Ebene gibt es einen einfachen Polygonzug, dessen Eckpunkte Endpunkte der Segmente sind, der kein Segment kreuzt, und der durch $\Omega(\log n)$ Segmente geht (von einem Endpunkt zum anderen). Dieses Resultat belegt, dass die Existenz Hamiltonscher Polygone in der Tat auch als strukturelles Ergebnis interessant ist.

Schliesslich wird noch eine Fragestellung aus dem Bereich der Kommunikations-Netzwerke behandelt: die Existenz von *sehnenfreien Pfaden* in Graphen. Auf einem sehnenfreien Pfad gibt es keine Kante

zwischen zwei Knoten, die nicht auch entlang des Pfades benachbart sind. Man könnte ebensogut sagen: Der von den Knoten des Pfades induzierte Teilgraph ist genau der Pfad selbst. Deshalb werden sehnenfreie Pfade auch oft als induzierte Pfade bezeichnet. Das Problem CP3V ist folgendes: Gegeben drei Knoten u, v und w eines Graphen G und eine natürliche Zahl k; gibt es einen sehnenfreien Pfad von u über v nach w in G, der aus höchstens k Knoten besteht?

Hier ist unser Ergebnis negativ: Wir zeigen, dass CP3V $W[1]$-vollständig ist. Das heisst, es ist unwahrscheinlich, dass es einen Algorithmus gibt, der es in Zeit polynomiell in der Grösse des Graphen löst, selbst wenn man eine beliebige Abhängigkeit (z.B. exponentiell) der Laufzeit von k erlaubt. Die Beweisidee lässt sich auch auf eine Reihe verwandter Probleme für gerichtete Graphen übertragen. Unter anderem ist es ein $W[1]$-vollständiges Problem, für zwei Knoten s und t eines gerichteten Graphen zu entscheiden, ob es einen sehnenfreien gerichteten Pfad von s nach t gibt.

Acknowledgments

I thank my advisor Emo Welzl for giving me the opportunity to work in an excellent environment, for numerous insights, his constant support, and the freedom to follow my various research interests.

I am grateful to Erik Demaine for accepting to co-referee this thesis, for helpful comments, inspiring talks, and many discussions about movable blocks and other things.

Csaba Tóth, Erik Demaine, Yoshio Okamoto, Bettina Speckmann, Udo Adamy, Oswin Aichholzer, Giordana Beutler, Marty Demaine, Robert Haas, Susan Hert, Markus Holzer, Lutz Kettner, Joseph O'Rourke, Sylvain Pion, Michael Seel, József Solymosi, Miloš Stojaković, Emo Welzl, and Gerhard Wöginger shared their ideas, their knowledge, and their experience with me. Without them my research would have been much less productive and way less fun.

Bernd Gärtner, Joachim Giesen, Bettina Speckmann, and Csaba Tóth helped me a lot to improve this manuscript by commenting on preliminary versions.

I had a great time with the members of our research group over the past years: Udo Adamy, Christoph Ambühl, Artur Andrzejak, Alexander Below, Robert Berke, Johannes Blömer, Péter Csorba, Kaspar Fischer, Bernd Gärtner, Joachim Giesen, Matthias John, Gyula Károlyi, Lutz Kettner, Alexander May, Dieter Mitsche, Yoshio Okamoto, Samuele Pedroni, Leo Rüst, Sven Schönherr, Eva Schuberth, Ingo Schurr, Shakhar Smorodinsky, József Solymosi, Simon Spalinger, Bettina Speckmann, Bernhard von Stengel, Miloš Stojaković, Tibor Szabó, Csaba Tóth, Beat Trachsler, Falk Tschirschnitz, Uli Wagner, Emo Welzl, Frans Wessendorp, and Martin Will.

Franziska Hefti, Tanja Krenn, and Floris Tschurr were a great help in resolving all kinds of administrative issues.

Finally, and most importantly, I thank my family for their support and love, far beyond what I can express within these lines.

Contents

Chapter 1

Introduction

This thesis investigates questions related to the existence of certain paths and cycles in graphs. It naturally divides into two parts.

The first part discusses traversals of line segments in the Euclidean plane. The graphs considered in this context are geometric graphs. In a geometric graph each vertex corresponds to a point and each edge corresponds to a line segment in the plane.

The second part of this thesis is concerned with chordless paths and cycles in abstract graphs. In contrast to the first part, these graphs are defined only by their sets of vertices and edges and do not have any specific associated embedding. Note that in a path or cycle all vertices are required to be distinct.

In the following two sections we introduce the problems to be considered, provide some background information, and discuss a number of related problems and results. Section 1.3 summarizes our results and provides an outline of the remaining chapters.

1.1 Traversals of Line Segments in the Plane

The Euclidean Traveling Salesman Problem (ETSP) is one of the most prominent optimization problems. The input consists of a set of n points in the plane and the goal is to compute a tour that visits all points and has minimum Euclidean length. It is well known that com-

puting such a minimum tour is a hard, that is, NP-hard problem [72], while approximations of arbitrary fixed precision can be obtained in polynomial time [7, 62, 76].

More important for our purposes is the simple observation that if not all points are collinear then the solution to an ETSP instance is always a simple polygon, that is, a shortest tour is piecewise linear and does not have any self-intersection [37, 75]. In particular, for every set of n points in the plane (that are not all collinear) there exists a simple polygon through all the points, that is, a polygon whose vertices are exactly the n given points.

We are interested in a generalization of the above statement to line segments: Can a simple polygon traverse the segment endpoints without crossing any segment? Consider the example shown in Figure 1 for illustration.

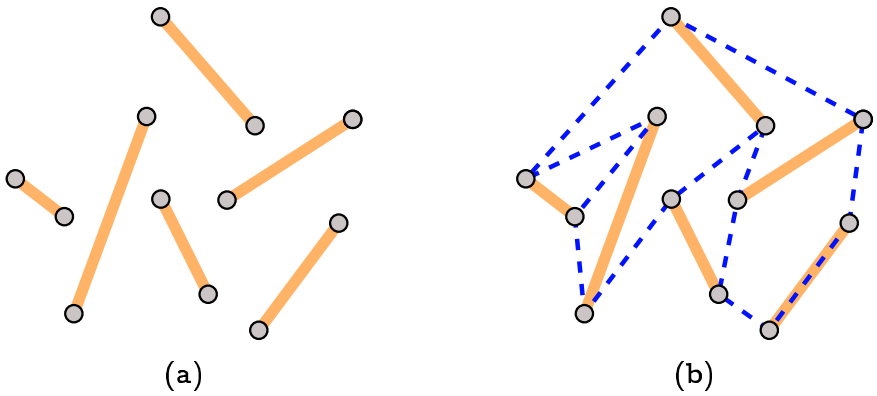


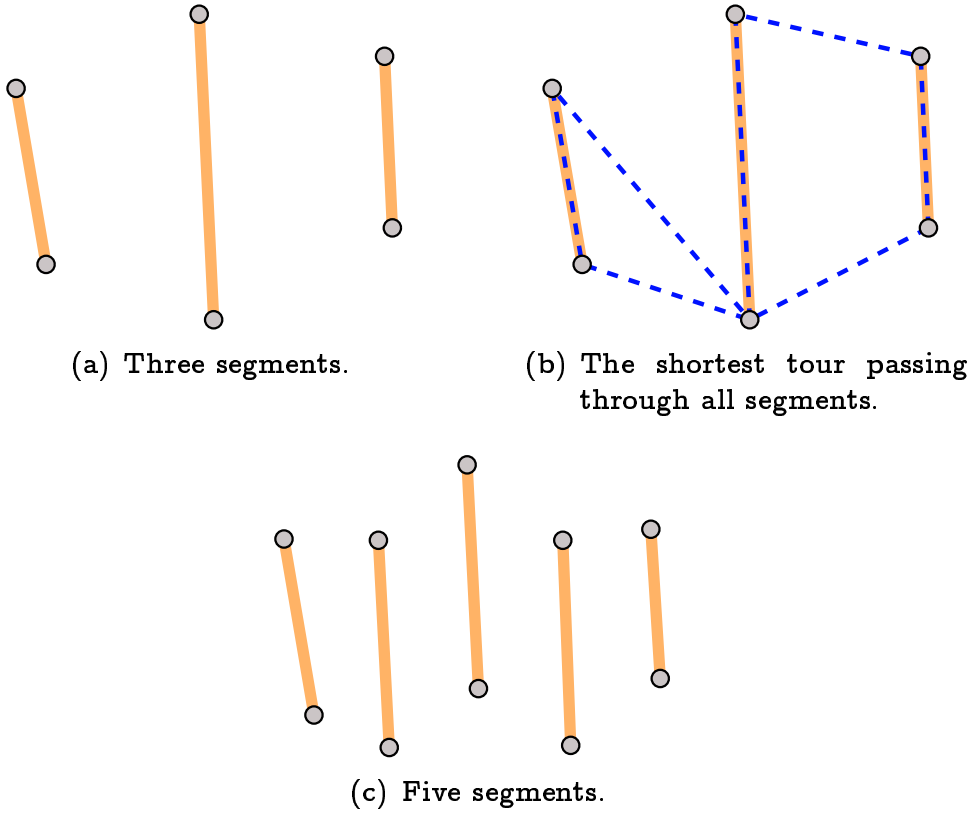
Figure 1: *A set of line segments and a simple traversal of the segment endpoints.*

There is a rich literature about simple traversals of objects in the plane. Some relevant results as well as several references to related problems are discussed in the following paragraphs.

Constrained Shortest Paths. It is common practice to model planar objects as (collections of) line segments, for example, in the context of geographic information systems. In particular, these objects may represent impenetrable obstacles, such as walls, rivers, mountains, or swamps. Therefore, we consider line segments to be obstacles that a path or tour must not cross. In general, the line segments may share endpoints or even intersect arbitrarily.

A well-studied question in this context is how to compute a shortest path between two given points in the presence of obstacles. In the special case where the obstacle segments form the boundary of a simple polygon (and their order along the polygon is given), one can find the shortest path in linear time [57]. For general line segments/polygonal domains Hershberger and Suri [47] gave an $O(n \log n)$ time (and space) algorithm.

Alternating Polygons. It is not completely clear what "traversing" or "visiting" a set of line segments means. As a first observation note that if visiting a segment is interpreted as passing through it from one endpoint to the other then there are simple examples for which the shortest tour crosses itself. In fact, for the three segments shown in Figure 2(a) there is no simple tour passing through all segments. Looking at Figure 2(a) one might think that disallowing chords (segments that are not edges of the convex hull but for which both endpoints are on the convex hull) may help. But as Figure 2(c) demonstrates, these are not the only obstructions that prevent a simple tour.



(a) Three segments.

(b) The shortest tour passing through all segments.

(c) Five segments.

Figure 2: *Sets which do not admit an alternating polygon.*

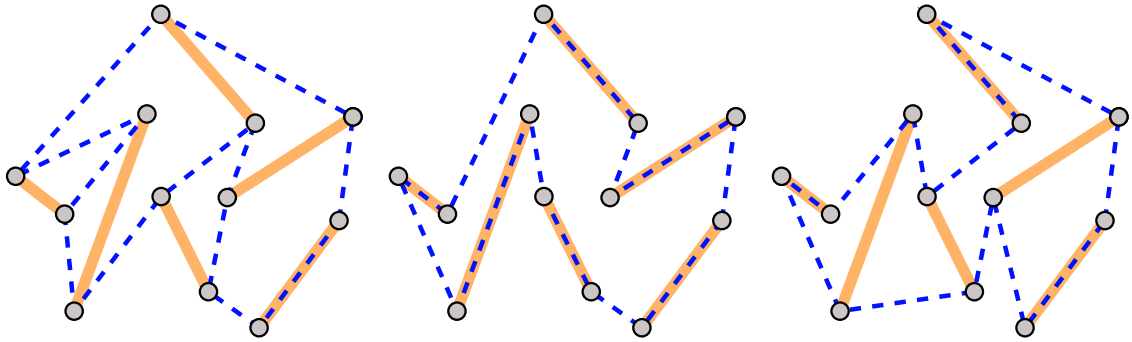
Rappaport [77] showed that it is an NP-complete problem to decide whether a given set of line segments can be completed to form the edges of a simple polygon whose vertices are the segment endpoints. We call such a polygon an *alternating polygon* because for disjoint segments exactly every other edge along the boundary of such a polygon must be one of the given segments.

Note that in Rappaport's reduction line segments are allowed to share endpoints, that is, he uses collections of polygonal paths; deciding on the existence of an alternating polygon for a set of disjoint line segments is not known to be NP-hard. In the special case where the given segments are convexly independent (for every segment at least one endpoint is on the convex hull boundary) an alternating polygon (if it exists) can be computed in linear time if the order of the segments along the convex hull is given [79]. Also the existence of a monotone alternating polygon for a set of line segments can be decided in polynomial time [8]. A simple polygon P is *monotone* if there is a line ℓ such that for any line ℓ' perpendicular to ℓ the intersection $\ell' \cap P$ is connected.

Hamiltonian Polygons. A natural relaxation of this apparently too strict definition of traversal is to require the polygon to pass through the segment endpoints only. This leads to the notion of Hamiltonian polygons: A *Hamiltonian polygon* for a set of line segments is a simple polygon whose vertices are the segment endpoints and for which no edge properly crosses any segment.

One main result of this thesis proves a conjecture of Mirzaian [61, 25]: A Hamiltonian polygon exists for every finite set of disjoint line segments that are not all collinear. Previously, this conjecture was known to hold for a few special classes of disjoint line segments only: for convexly independent segments [61], when no segment intersects the supporting line of any other segment [69], and for unit length segments whose endpoints have integer coordinates [69].

Moreover, our proof provides an algorithm to construct a Hamiltonian polygon for n disjoint line segments in $O(n^2)$ time. Given a lower bound of $\Omega(n \log n)$ (see the paragraph on encompassing trees below), a natural open problem remains to close this gap.



(a) Hamiltonian. (b) Alternating. (c) Circumscribing.

Figure 3: *Different classes of polygons traversing a set of line segments.*

Circumscribing Polygons. Another plausible interpretation of traversal asks for slightly more: a simple polygon whose vertices are the segment endpoints and that contains all segments in its interior (as a closed subset of $\mathbb{R}^2$). Such a polygon is known as a *circumscribing polygon* for the segments. An example illustrating the differences between the different types of traversal polygons is shown in Figure 3.

Mirzaian [61] proved that a circumscribing polygon exists for disjoint line segments that are convexly independent. He conjectured that a circumscribing polygon exists for every finite set of disjoint line segments, but this conjecture was quickly refuted by Urabe and Watanabe [86]: they constructed a set of seventeen disjoint line segments which do not admit a circumscribing polygon. The even smaller counterexample shown in Figure 4 is due to Grünbaum [43].

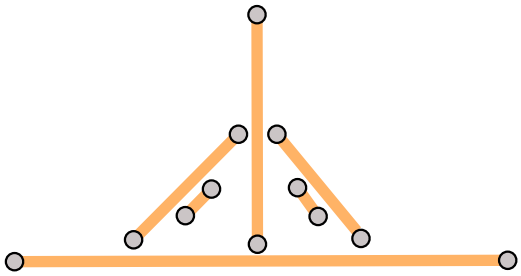


Figure 4: *Six segments which do not admit a circumscribing polygon.*

On the other hand, Pach and Rivera-Campo [71] showed that for every set of n disjoint line segments in general position (no three end-

points are collinear) there is a subset of size $\Omega(n^{1/3})$ segments for which a circumscribing polygon exists.

Segment Endpoint Visibility Graphs. All these polygons can also be interpreted as spanning subgraphs of the so-called *segment endpoint visibility graph*. The vertices of this graph are the segment endpoints and two of them are connected by an edge if and only if the corresponding line segment is either one of the input segments or it does not cross any input segment; Figure 5 shows an example.

The third conjecture of Mirzaian [61] is that for any finite set of disjoint line segments in the plane — not all collinear — the endpoint visibility graph is Hamiltonian. Clearly a circumscribing polygon is also a Hamiltonian polygon, and a Hamiltonian polygon gives a Hamiltonian cycle in the visibility graph. Hence, our theorem regarding the existence of Hamiltonian polygons for disjoint segments also settles Mirzaian's third conjecture. On the other hand, we show that the disjointness condition cannot be dropped from the statement: If the segments may share endpoints, their endpoint visibility graph is not necessarily Hamiltonian.

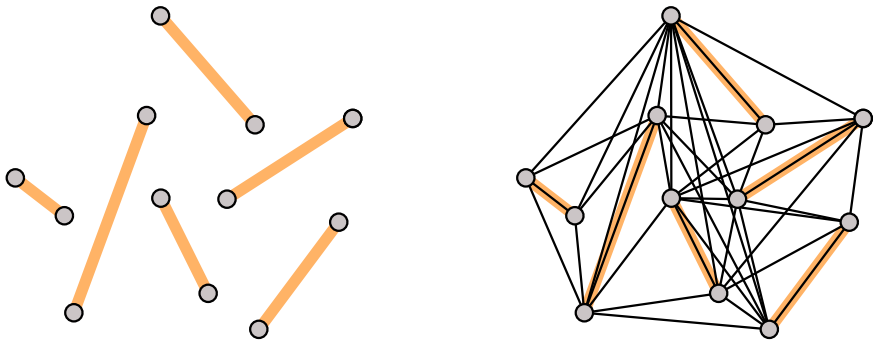


Figure 5: *A set of segments and their endpoint visibility graph.*

Apart from their obvious applications in illumination, rendering, and guarding, visibility graphs are also an important structure in many motion planning problems (cf. [63]), as the visibility graph contains all shortest paths between obstacle vertices. For polygonal obstacles consisting of n line segments, the visibility graph can be computed in worst case optimal quadratic time (and space) [90] or output-sensitive in $O(n \log n + k)$ time [42], even with $O(n)$ (working) space [74], where

k is the number of edges in the visibility graph.

If the segments are on the boundary of a convex polygon then the visibility graph is a complete graph on $2n$ vertices, so k may be quadratic in n. On the other hand, $5n - 4$ is a tight lower bound on k that is obtained if all segments are parallel and all endpoints are in convex position [82]. It is not known whether segment endpoint visibility graphs can be recognized efficiently, except for the (small) subclass of planar segment endpoint visibility graphs for which Everett et al. [33] gave a linear time recognition algorithm.

If the segments form the boundary of a simple polygon, the visibility graph can be computed in $O(n + k)$ time [46]; in this case also a more compact representation of size $O(n \log^3 n)$ may be obtained as a union of cliques and bipartite cliques [1]. However, for n disjoint line segments the smallest clique cover may be of size $\Omega(n^2/\log^2 n)$ [1].

Alternating Paths. As discussed above, there are sets of disjoint line segments that do not admit an alternating polygon. But maybe — as for circumscribing polygons — one can always find a "large" subset of segments for which an alternating polygon exists? It is not hard to see that this may not be the case. Consider a set of parallel segments whose endpoints form a convex polygon; no subset of more than two of these segments admits an alternating polygon.

On the other hand, there is always an alternating Hamiltonian path in the visibility graph of parallel line segments. (In an *alternating path* every other edge is one of the given segments.) So maybe one can always find an alternating path through "many" segments? Indeed, we show that for any set of n disjoint line segments the visibility graph contains an alternating path through roughly $\log n$ segments. This bound is asymptotically tight: For any $k \in \mathbb{N}$ there exists an $n \geq k$ and a set of n disjoint line segments whose visibility graph does not contain an alternating path through more than $4 \log n$ segments. To close the gap between these bounds and, ideally, to determine the "right" constant remains an open problem.

Encompassing Trees. An immediate consequence of the existence of a Hamiltonian polygon is that for any set of n disjoint line segments there is an *encompassing tree*. An encompassing tree is a planar embedding of a tree in which all segments are edges; see the example shown in

Figure 6(a). Indeed, a Hamiltonian polygon together with all segment edges forms a planar spanning subgraph of the visibility graph with maximum vertex-degree three. For this graph one can easily compute a spanning tree that contains all segments as edges and hence forms a binary encompassing tree for the segments. (Take an arbitrary cycle in the graph and remove a non-segment edge from it. As the segments are disjoint, any cycle contains a non-segment edge.)

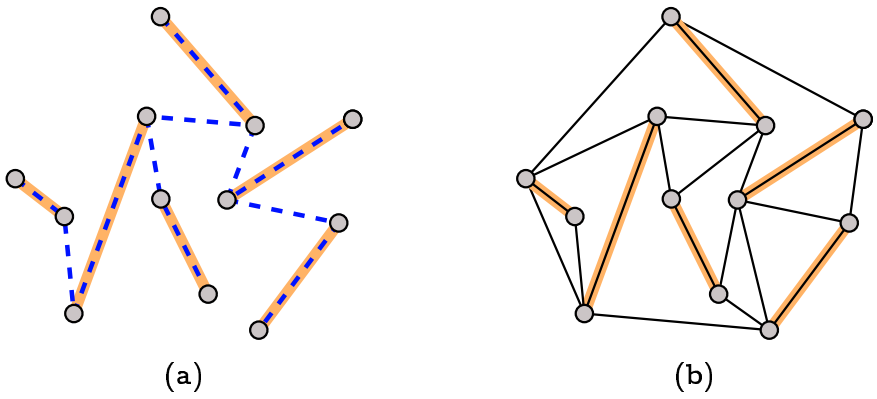


Figure 6: *A (pointed) binary encompassing tree and a minimum pseudotriangulation for a set of disjoint line segments.*

The existence of an encompassing tree for disjoint line segments has been established by Bose and Toussaint [16], with a degree bound of seven. They also showed that this bound is tight in the case of minimum weighted encompassing trees, where each edge is weighted with the Euclidean distance of both endpoints. Later Bose, Houle, and Toussaint [15] improved the general bound to three and gave an algorithm to compute a binary encompassing tree for n segments in $O(n \log n)$ time. They also showed that the runtime is optimal in the algebraic computation tree model. As a binary encompassing tree can be obtained from a Hamiltonian polygon in linear time, this lower bound carries over to the problem of constructing a Hamiltonian polygon for a set of disjoint line segments.

Recently Hoffmann, Speckmann, and Tóth [48] strengthened these results by showing that any set of disjoint line segments admits a *pointed* binary encompassing tree. A geometric graph is pointed if and only if for every vertex all incident edges lie in a closed halfplane that is bounded by a line through the vertex. Pointedness is an important property of so-called minimum pseudotriangulations. A *pseudotrian-*

gle is a simple polygon with exactly three convex vertices; in particular, any triangle is a pseudotriangle. A *minimum pseudotriangulation* is a partition of some domain (for example, a polygon) into a minimum number of pseudotriangles; see Figure 6(b) for an example.

The existence of a pointed binary encompassing tree implies that there exists a constrained minimum pseudotriangulation for the segments (that is, a minimum pseudotriangulation in which all segments appear as edges) whose maximum vertex degree is bounded by a constant, independently of the number of segments [2]. Observe that the analogous statement for triangulations is false: For some sets of segments (even points) every constrained triangulation has a vertex of linear degree. Also, there are triangulations that do not contain a pointed spanning tree for their vertices as a subgraph [3].

The Number of Simple Polygonizations. Interestingly, it is not known whether the number of simple polygons through a given point set (that is, whose vertices are exactly the given points) can be computed in polynomial time [64, 70]. Currently, the best known estimates for the maximum number of simple polygons through n points are a lower bound of 4.642^n [40] and an upper bound of 199^n (obtained by combining an upper bound of 3.37^n on the number of simple cycles in a planar graph [5] with an upper bound of 59^n on the number of triangulations for a set of n points [81]).

Extensions to Higher Dimension. The statement about the existence of a simple polygon through a given set of points generalizes to three dimensional space in the following way: For every finite set of points in $\mathbb{R}^3$ not all of which are coplanar there is a simple (sphere-like, that is, genus zero) polyhedron whose vertices are exactly the given points [43, 53].

Csima and Ralston [24] showed that between any two points of a finite point set $P \subset \mathbb{R}^d$ there exists a simple Hamiltonian path (in the complete geometric graph on P), unless the points are arranged in one of six forbidden types of configurations. (In all of these exceptional configurations many points are collinear.)

Simple Hamiltonian Cycles in Dense Geometric Graphs. The following problem is due to Perles [73]: How many edges can one remove from any

complete geometric graph on n points (in general position) such that the resulting graph still always contains a simple Hamiltonian cycle? Černý et al. show that if $o(\sqrt{n})$ edges are removed then the resulting graph always contains a simple Hamiltonian cycle [18]. They also show that if the set of removed edges corresponds to a perfect matching then the resulting graph always contains a simple Hamiltonian cycle. If we interpret this result in terms of disjoint line segments, removing the edges of a perfect matching corresponds to disallowing the segments as edges. But the resulting Hamiltonian cycle is not a Hamiltonian polygon for the set of line segments as defined above; it is only a Hamiltonian cycle for the segment endpoints and its edges may cross the line segments arbitrarily.

Long Hamiltonian Cycles. Similar to the ETSP where the objective is to find a minimum tour, researchers considered the corresponding maximization problem MAX-ETSP of computing a tour of maximum length through the given input points. Surprisingly, the complexity of MAX-ETSP in the plane is still open [63], while the problem is known to be NP-hard starting from dimension three [9]. Another interesting fact is that the problem can be solved in polynomial time for any fixed dimension if the Euclidean norm is replaced by the rectangular norm (or, more generally, any norm defined by a finite polyhedron) [9]. This is in strong contrast to the minimization problem which is NP-hard for any L_p-norm starting from dimension two [54].

Another difference between (MIN-)ETSP and MAX-ETSP is that a maximum tour usually contains self-crossings. Thus, requiring the tour to be simple is indeed a restriction. While it seems that the problem of computing a maximum simple tour should be at least as hard as MAX-ETSP, this problem is not known to be NP-hard, either. Alon et al.[4] showed that for any finite set of points in the plane a non-crossing Hamiltonian path whose length is within a factor of $1/\pi$ of the longest (possibly self-crossing) Hamiltonian path can be computed in polynomial time. They also gave a simple example — n points evenly distributed on the unit circle — demonstrating that one cannot hope for a similar constant ratio in the case of Hamiltonian cycles: in this case the length of the longest simple cycle is less than 2π, but the length of the longest self-crossing Hamiltonian cycle is linear.

Hamiltonian Triangulations. Another well-studied problem is the existence of Hamiltonian cycles in triangulations. It appeared in the context of curve reconstruction where one has to construct a closed curve through a given set of points. The first algorithms to attack this problem started by building a Delaunay triangulation of the input points (which tends to include edges between points close to each other) and then find a Hamiltonian cycle in the triangulation. Naturally, the question was raised whether such a Hamiltonian cycle always exists. The answer is negative [26]; in fact, it is an NP-complete problem to decide whether a given Delaunay triangulation is Hamiltonian [28].

On the other hand, a sufficient condition for the existence of a Hamiltonian cycle in a triangulation is provided by a classical theorem of Whitney [91]: for maximally planar graphs (all faces are triangles) the absence of a separating triangle (a triangle that does not form the boundary of a face) implies that the triangulation is Hamiltonian. Dillencourt [27] discusses extensions of Whitney's theorem to triangulations where the outer face is not a triangle. He shows that a triangulation is Hamiltonian if it does not have a separating triangle and if in the subgraph induced by the vertices of the outer face (the so-called *boundary graph*) no face is bounded by more than three chords (edges not bounding the infinite face). Observe that properly placed line segments may generate a linear number of chords in the boundary graph; hence, Dillencourt's Theorem does not (immediately) imply the existence of a Hamiltonian polygon for disjoint line segments. But Dillencourt's Theorem can be used to show that the visibility graph of n congruent discs in the plane is Hamiltonian [78].

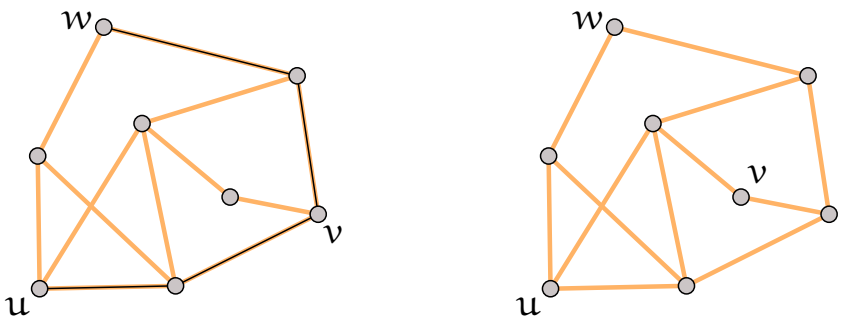
Unfortunately, the term *Hamiltonian triangulation* is also used to denote a different property, the presence of a Hamiltonian path in the dual of the triangulation. This question is motivated by an application from computer graphics. When triangulation data is sent to a rendering engine, in order to minimize the amount of communication it is desirable to order the triangles in such a way that any two consecutive triangles share an edge. Then for each triangle only one new vertex has to be transmitted. (To be precise, one more bit is needed to indicate the direction in which to continue.) Clearly such an ordering of the triangles corresponds to a Hamiltonian path in the dual graph of the triangulation. For any set of n points [6] and any simple polygon on n vertices [66] a Hamiltonian triangulation — if it exists — can be constructed in $O(n \log n)$ time. But deciding the existence of a

Hamiltonian triangulation for polygons with holes is NP-complete [6].

1.2 The Parametric Complexity of Chordless Paths

A *chordless path* in a graph G is a path for which no two vertices are connected by an edge that is not in the path. Alternatively, one could say that the subgraph induced by the vertex set of the path in G is the path itself. Hence, chordless paths are also known as *induced paths*.

The problem CP3V ("Chordless Path through Three Vertices") is to decide for three given vertices u, v, w of a graph whether there exists a chordless path from u via v to w. Figure 7 shows an example. This problem was introduced by Robert Haas who encountered it in the context of service deployment in communication networks [44].



(a) A chordless path. (b) No chordless path.

Figure 7: *Vertices u, v, w for which a chordless path from u via v to w exists or does not exist.*

The motivation is the following: For a given network (modeled as a graph) one is given two vertices u and w and one has to find a path between these vertices such that all vertices on the path satisfy certain criteria (one may think of bandwidth or support for certain protocols as an example). The list of criteria is typically long and the capabilities of the vertices vary over time; that is, we do not know which vertices of the graph satisfy those criteria for any particular request. Hence we may also look at this problem as a search for a path in an unknown induced subgraph of a given graph.

The computation consists of rounds in each of which a subset of vertices is queried. At the end of each round we know which of the queried vertices satisfy the given criteria. The goal is to minimize the

number of rounds and at the same time the number of vertices queried. Clearly these two objectives are contradictory and we did not specify how to balance them, that is, the overall objective is not well defined.

But the following simple observation allows to potentially discard a number of vertices: As the (unknown) subgraph of admissible vertices is an induced subgraph, we may restrict our search to chordless paths. That is, a vertex that does not lie on any chordless path between u and w does not need to be queried. For suppose that we found a path P from u to w for which all vertices fulfill the given list of requirements. If P has a chord then this chord may be used as a shortcut to produce a shorter path from u to w for which all vertices satisfy the requirements (because they were vertices of the original path P). Observe that simply the fact that there exists a path that is shorter than P does not help, as long as it is not clear that all vertices on this path satisfy the given requirements.

Indeed, experiments have shown that this discard strategy is effective: For typical Internet router-level topologies, a large fraction of vertices do not belong to chordless paths. Depending on the topology analyzed, a conservative estimate of the average percentage of vertices that are not on a chordless path for random pairs of source and destination vertices varies from 9% to 48% of the total number of vertices in the topology [44].

Unfortunately, as far as an efficient implementation of the strategy is concerned, our result is negative: It turns out that CP3V is NP-complete as a consequence of a theorem by Fellows [35]. However, from an application point of view one might be willing to accept an algorithm that is exponential in some parameter that is "typically" small; in this case, the number of vertices on the path is a natural parameter. Hence, let us add a positive integer k to the input of CP3V and rephrase it to ask for a chordless path from u via v to w of size at most k, where the size of a path is defined as its number of vertices.

But once again it turned out that an efficient algorithm for the parameterized version is rather unlikely to exist: We prove the problem to be $W[1]$-complete with respect to k. This also implies [10, 21] that there is probably no PTAS to compute a $(1+\varepsilon)$-approximation for the shortest chordless path from u via v to w in time bounded by an arbitrary function in ε but polynomial in the size of the graph.

The reduction extends to a number of related problems about chord-

less paths and cycles. In particular, deciding on the existence of a single directed chordless (u, w)-path in a digraph is also $W[1]$-complete with respect to the size of the path.

In the following paragraphs we summarize a number of related results and provide a brief introduction into parameterized complexity.

Chordless Paths and Cycles. The following problem is closely related to Cp3v.

Many Chordless Paths through Two Vertices (Mcp2v): *Given an undirected graph* $G = (V, E)$*, positive integers* k *and* ℓ*, and two distinct vertices* $s, t \in V$*, is there a set* $U \subseteq V$ *of at most* k *vertices such that the subgraph induced by* U *in* G *is a disjoint union of* ℓ *chordless* (s, t)*-paths?*

The unparameterized version of the above problem (no restriction on the size of U) was shown to be Np-complete by Fellows [35], already for $\ell = 2$ where it asks for a chordless cycle through s and t. We show that this problem is equivalent to the unparameterized version of Cp3v under polynomial time reductions, that is, the unparameterized version of Cp3v is Np-complete. Furthermore, these reductions can also be considered parameterized reductions between Cp3v and Mcp2v. In particular, it is a $W[1]$-complete problem to decide whether there exists a chordless cycle of length k through two given vertices of a graph.

The hardness results for Cp3v and 2cp2v rely on graphs that contain many vertex-disjoint (s, t)-paths. It is not difficult to see that in planar graphs the existence of four vertex-disjoint (s, t)-paths basically[1] implies the existence of a chordless (s, v, t)-path. This argument immediately gives an $O(3^k n^c)$ time algorithm for Cp3v in planar graphs, for some constant $c \in \mathbb{N}$: Branch on the at most three vertices of a minimum (s, t)-cut. But the more interesting question is whether Mcp2v is polynomial for planar graphs. This was answered in the affirmative for every fixed ℓ by McDiarmid et al. [59, 60].

If in Mcp2v we ask for vertex-disjoint paths only instead of requiring all paths to be jointly chordless, the problem is polynomial for general graphs and every fixed ℓ, even for arbitrary source-target pairs (s_i, t_i), $1 \leq i \leq \ell$ [80]. But it remains Np-complete if ℓ is considered part of the input [56].

[1] Except for some basic cases which can easily be solved separately.

Deciding whether a graph contains a chordless path of size at least k is one of the classical NP-complete problems (GT23 in [41]). Bienstock [13, 14] listed several other NP-complete problems related to chordless paths:

- Does a graph contain a chordless path of odd (even) size between two specified vertices?
- Does a graph contain a chordless cycle of odd (even) size (> 3) through a specified vertex?
- Does a graph contain a chordless path of odd (even) size between any two vertices?

Note that these results do not imply the hardness of deciding whether there exists **any** chordless path/cycle of odd/even size (> 3) in a graph. This question is still open; see the discussion below.

Perfect Graphs. Chordless cycles of size at least four are also called *holes*. They are tightly connected to Berge's strong perfect graph conjecture [11], whose proof has recently been announced by Chudnovsky et al. [23]. In a perfect graph, the maximum number of pairwise adjacent vertices (*clique number*) for each induced subgraph is equal to the minimum number of colors needed to color the vertices in such a way that any two adjacent vertices receive distinct colors (*chromatic number*). According to the perfect graph conjecture, a graph is perfect if and only if it is Berge, that is, it contains neither an odd hole nor the complement of an odd hole.

Hence, a polynomial time algorithm to decide whether there exists any odd hole in a given graph would immediately imply that perfect graphs can be recognized in polynomial time. Interestingly, no such algorithm to detect odd holes is known, although there is a polynomial time algorithm to decide whether a graph is Berge [22], even independent of the strong perfect graph conjecture. Also, if the restriction to an odd number of vertices is omitted, the presence of holes can be detected in polynomial time: for holes on at least four vertices this is the well-studied recognition problem for chordal graphs [58, 85]. The problem of detecting holes on at least five vertices has recently been addressed by Nikolopoulos and Palios [68].

Parameterized Complexity. To cope with the apparent computational intractability of NP-hard problems, attempts were made to analyze more closely which parts or aspects of the input render a particular problem hard. A prototypical example is *Vertex Cover*, which asks for a set C of at most k vertices from a given graph G on n vertices such that for each edge at least one endpoint is in C. The trivial observation that for any edge at least one of the two incident vertices has to be in C, leads to an $O(2^k n)$ time algorithm: Choose an arbitrary edge and branch on the two possibilities, in both cases removing one vertex and all incident edges from the graph. Hence, the intractability of Vertex Cover is connected to the number k of vertices in the cover rather than to the size of the graph G. One says that Vertex Cover is *fixed-parameter-tractable* (FPT) with respect to the parameter k because there is an algorithm that runs in $f(k)p(n)$ time for an arbitrary (typically exponential) function f and a polynomial p. Such an algorithm is said to be an FP (fixed-parameter) algorithm.

Naturally, there are also problems for which it is not known whether their complexity can be attributed to one particular parameter in this way. Moreover, similar to the classical complexity classes, there are classes of parameterized problems that are hard in the sense that if there is a fixed parameter algorithm for any of them, then all of them are FPT. The most important such class is called $W[1]$, which can be described in terms of the following "canonical" problem.

Weighted q-CNF-Satisfiability: *Given two positive integers q and k, and a boolean formula F in conjunctive normal form such that each clause contains at most q literals, is there a satisfying assignment for F with at most k variables set to true?*

A problem P parameterized by k is said to be m-*reducible* to a problem P' parameterized by k' if there is a computable function h that maps an instance (x, k) of P to an instance (x', k') of P' such that

- (x, k) is a yes-instance of P if and only if (x', k') is a yes-instance of P',
- $k' = g(k)$,
- and x' can be computed in time $f(k)p(|x|)$,

where f and g are arbitrary functions and p is a polynomial. Then $W[1]$ is defined as the class of parameterized problems that can be m-

reduced to Weighted q-CNF-Satisfiability for some constant q. Finally, a problem is $W[1]$-*hard* if every problem in $W[1]$ can be m-reduced to it. A problem that is both $W[1]$-hard and in $W[1]$ is called $W[1]$-*complete*.

At this point, we refer the interested reader to the literature for more in-depth information about parameterized complexity. The book of Downey and Fellows [30] provides a thorough treatment of complexity-theoretic aspects, whereas the survey of Niedermeier [67] focuses more on algorithms.

1.3 Results and Outline

Chapter 2 introduces basic structures, concepts, and notation that are used throughout this thesis.

In Chapter 3 we prove the existence of a Hamiltonian polygon for any set of disjoint line segments in the plane in which not all segments are collinear. The proof is algorithmic and yields an $O(n^2)$ time algorithm to construct a Hamiltonian polygon for a set of n disjoint line segments. This result was obtained in collaboration with Csaba Dávid Tóth; it was presented at CCCG 2001 [49] and then invited to a special issue of *Computational Geometry: Theory and Applications* [52]. We also give an example that the above statement is not true in general if the segments may share endpoints.

Chapter 4 uses the existence of a Hamiltonian polygon to resolve another long-standing open problem [25, 87, 88] regarding *alternating paths*: for every set of n disjoint line segments the endpoint visibility graph contains an alternating path (a path that consists of segment edges and non-segment edges in alternating order) that passes through at least $\Omega(\log n)$ segments. This bound is asymptotically tight in the sense that there are families of disjoint line segments which do not admit an alternating path through more than $O(\log n)$ segments. This result was presented at EWCG 2002 [50] as a joint work with Csaba Dávid Tóth; the full paper appeared one year later in *Information Processing Letters* [51].

Chapter 5 settles the parametric complexity of CP3V and a few related problems by proving them $W[1]$-complete. Recall that CP3V is to decide for three given vertices u, v, w of a graph whether there exists a chordless path from u via v to w. Among the other problems are the

following.

- Is there a chordless cycle of size at most k through two given vertices?
- Is there a directed chordless path of size at most k between two given vertices of a directed graph?
- Is there a directed chordless cycle of size at most k through a given vertex of a directed graph?

These results were presented at IWPEC 2004 [45] as a joint work with Robert Haas.

Chapter 2

Basics and Notation

This chapter defines the objects of our interest and derives some of their properties. It also lists some basic concepts and notation that will be used throughout this work.

2.1 Graphs

An (undirected) graph $G = (V, E)$ is defined on a set V of *vertices*. Unless explicitly stated otherwise, V is always finite. Vertices are associated to each other through *edges* which are collected in the set $E \subseteq \binom{V}{2}$. The two vertices defining an edge are termed *adjacent* to each other and *incident* to the edge.

For a vertex $v \in V$, denote by $N_G(v)$ the *neighborhood* of v in G, that is, the set of vertices from G that are adjacent to v. Similarly, for a set $W \subset V$ of vertices define $N_G(W) := \bigcup_{w \in W} N_G(w)$. The *degree* $\deg_G(v)$ of a vertex $v \in V$ is the size of its neighborhood, that is, the number of edges from E incident to v. The subscript is often omitted when it is clear to which graph it refers to.

Two graphs $G = (V, E)$ and $H = (U, W)$ are *isomorphic* if and only if there is a bijection $\phi : V \to U$ such that $\{u, v\} \in E \iff \{\phi(u), \phi(v)\} \in W$. Such a function ϕ is called an *isomorphism* between G and H. The structure of isomorphic graphs is identical and often we do not distinguish between them when looking at them as graphs.

For a graph G denote by $V(G)$ the set of vertices and by $E(G)$ the set of edges. A graph $H = (U, F)$ is a *subgraph* of G if and only if $U \subseteq V$ and $F \subseteq E$. In case that $U = V$ the graph H is a *spanning* subgraph of G. For a set $W \subseteq V$ of vertices denote by $G[W]$ the *induced subgraph* of W in G, that is, the graph $(W, E \cap \binom{W}{2})$. For $F \subseteq E$ let $G \setminus F := (V, E \setminus F)$. Similarly, for $W \subseteq V$ let $G \setminus W := G[V \setminus W]$. In particular, for a vertex or edge $x \in V \cup E$ we write $G \setminus x$ for $G \setminus \{x\}$. The *union* of two graphs $G = (V, E)$ and $H = (W, F)$ is the graph $G \cup H := (V \cup W, E \cup F)$.

Graph Traversals A *walk* in G is a sequence $W = (v_1, \ldots, v_k)$, $k \in \mathbb{N}$, of vertices such that v_i and v_{i+1} are adjacent in G, for all $1 \leq i < k$. The vertices v_1 and v_k are referred to as the walk's *endpoints*, the other vertices are called *interior*. A walk with endpoints v_1 and v_k is sometimes referred to as a *walk between* v_1 and v_k. For a walk W denote by $V(W)$ its set of vertices and by $E(W)$ its set of edges (pairs of vertices adjacent along W). We say that W *visits* the vertices and edges in $V(W) \cup E(W)$. A walk for which both endpoints coincide, that is, $v_1 = v_k$, is called *closed*. Otherwise the walk is *open*.

The *size* $|W|$ of a walk $W = (v_1, \ldots, v_k)$ is defined as its number of vertices, counting multiplicities: for an open walk it is $|W| = k$; if W is closed then v_1 and v_k are counted as one appearance of the vertex, that is, $|W| = k - 1$. A walk of size one is said to be *trivial*. An *odd walk* is a walk whose size is odd. Analogously, an *even walk* is a walk whose size is even. For two walks $U = (u_1, \ldots, u_k)$ and $W = (v_1, \ldots, v_\ell)$, for $k, \ell \in \mathbb{N}$, with $u_k = v_1$ denote the *concatenation* of U with W by $U \cdot W := (u_1, \ldots, u_k, v_2, \ldots, v_\ell)$.

If a walk uses each edge of G at most once, it is a *trail*. A closed walk that visits each edge and each vertex at least once is called a *tour* of G. An *Euler tour* is both a trail and a tour of G, that is, it visits each edge of G exactly once. A graph that contains an Euler tour is termed *Eulerian*.

If all vertices $v_1, \ldots, v_k$ of a closed walk W are distinct except for $v_1 = v_k$ then W is a *cycle* of size $k - 1$. If all vertices $v_1, \ldots, v_k$ of a walk W are distinct, W is a *path* of size k. A *Hamilton cycle (path)* is a cycle (path) that visits every vertex of G. A graph that contains a Hamilton cycle is termed *Hamiltonian*.

Two trails are *edge-disjoint* if and only if they do not share any edges. Two paths are called *vertex-disjoint* if and only if they do

not share any vertices except for possibly common endpoints. For two vertices $s, t \in V$ any path with endpoints s and t is called an (s,t)-path. More generally, if vertices s, v, and t appear on path P in this order, we call P an (s,v,t)-path.

A set $I \subseteq V$ of vertices is an *independent set* in G, if E does not contain edges between any two vertices of I. A path P in G is called *chordless* if and only if $V(P)$ is an independent set in $G \setminus E(P)$. An alternative equivalent definition would be to call a path P in G chordless if and only if $G[V(P)] = P$. Hence such paths are also known as *induced paths*.

A *subwalk* of a walk $W = (v_1, \ldots, v_k)$ is a contiguous subsequence $(v_i, \ldots, v_j)$ of W, for $1 \leq i \leq j \leq k$. If W is closed then also subsequences of the type $(v_i, \ldots, v_k, v_2, \ldots, v_j)$, for $2 \leq j \leq i \leq k$, are considered to be subwalks of W. A subwalk that is a trail or path is also referred to as *subtrail* or *subpath*, respectively.

Define an equivalence relation "$\sim$" on V by setting $a \sim b$ if and only if there is a path between a and b in G. The equivalence classes with respect to "$\sim$" are called *components* of G and their number is denoted by $\omega(G)$. G is *connected* if $\omega(G) = 1$ and *disconnected*, otherwise.

A vertex $v \in V$ of a connected graph $G = (V, E)$ is a *cut-vertex* of G if and only if $G \setminus v$ is disconnected. A graph that does not contain any cut-vertex is *2-connected*. Similarly an edge $e \in E$ of a connected graph $G = (V, E)$ is a *cut-edge* of G if and only if $G \setminus e$ is disconnected. A graph that does not contain any cut-edge is *2-edge-connected*.

Special Classes of Graphs A graph with a maximum number of edges, that is, $(V, \binom{V}{2})$, is called a *clique*. Up to isomorphism there is only one clique on n vertices; it is referred to as the *complete graph* K_n, $n \in \mathbb{N}$. A graph whose vertex set can be partitioned into at most two independent sets is *bipartite*. An equivalent characterization states that a graph is bipartite if and only if it does not contain any odd cycle. The bipartite graphs with a maximum number of edges (unique up to isomorphism) are the *complete bipartite graphs* $K_{m,n}$, for $m, n \in \mathbb{N}$. They consist of two disjoint independent sets of size m and n, respectively, and all mn edges in between.

A *forest* is a graph that does not contain any cycle. A connected forest is called *tree* and its *leaves* are the vertices with exactly one neighbor. Every connected graph contains a spanning subgraph which

is a tree, a so called *spanning tree*.

2.2 Directed Graphs

An directed graph or, for short, *digraph* $D = (V, E)$ is defined on a set V of *vertices*. Unless explicitly stated otherwise, V is always finite. The set E consists of ordered pairs of vertices, that is, $E \subseteq V^2$. The elements of E are referred to as *arcs*. An arc $(u, v) \in E$ is said to be directed from its *source* u to its *target* v. For $(u, v) \in E$ we also say "there is an arc from u to v in D". Usually, we consider *loop-free* graphs, that is, arcs of the type (v, v), for some $v \in V$, are not allowed.

The *in-degree* $\deg_D^-(v) := |\{(u, v) | (u, v) \in E\}|$ of a vertex $v \in V$ is the number of *incoming* arcs at v. Similarly, the *out-degree* $\deg_D^+(v) := |\{(v, u) | (v, u) \in E\}|$ of a vertex $v \in V$ is the number of *outgoing* arcs at v. Again the subscript is often omitted when the graph under consideration is clear from the context.

From any undirected graph G one can obtain a digraph on the same vertex set by specifying a direction for each edge of G. Each of these $2^{|E(G)|}$ different digraphs is called an *orientation* of G. Similarly every digraph $D = (V, E)$ has an *underlying* undirected graph $G = (V, \{\{u, v\} \mid (u, v) \in E \text{ or } (v, u) \in E\})$. Hence most of the terminology for undirected graphs carries over to digraphs.

A *directed walk* in a digraph D is a sequence $W = (v_1, \ldots, v_k)$, for some $k \in \mathbb{N}$, of vertices such that there is an arc from v_i to v_{i+1} in D, for all $1 \leq i < k$. In the same way we define *directed trails*, *directed paths*, *directed cycles*, and *directed tours*.

2.3 Geometry

Most of the following takes place in the Euclidean Plane $\mathbb{R}^2$. We are particularly interested in the following types of subsets of $\mathbb{R}^2$.

An element of $\mathbb{R}^2$ is referred to as a *point*.

A *line* ℓ is a one-dimensional affine subspace of $\mathbb{R}^2$. It is uniquely determined by two distinct points p and q as

$$\ell = \{p + \lambda(q - p) \mid \lambda \in \mathbb{R}\} .$$

Removal of any single line ℓ disconnects $\mathbb{R}^2$ into two components. These are the *open halfplanes* defined by ℓ. Their respective closure or, equivalently, union with ℓ forms a *closed halfplane*.

A line ℓ is disconnected into two components by removal of any single point $p \in \ell$. Their respective closure or, equivalently, union with p forms a *ray* emanating from p along ℓ. For any point $q \in \ell \setminus \{p\}$ we also denote these rays by

$$\overrightarrow{pq} := \{p + \lambda(q - p) \mid \lambda \in \mathbb{R} \text{ and } \lambda \geq 0\} .$$

Any non-empty, compact and connected subset of a line is a *line segment* or *segment*, for short. For a segment s let $V(s)$ denote the set of its *endpoints*, that is, those points whose removal does not disconnect s. Note that this definition allows segments that consist of a single point only; we call such segments *degenerate*.

For any ray and any non-degenerate segment there is a unique line containing it. This line is referred to as the *underlying line* of the ray or segment. We say that two sets $A, B \subset \mathbb{R}^2$ are *collinear* if and only if there exists a line that contains both. Any two distinct points $p, q \in \mathbb{R}^2$ define a unique line segment

$$\overline{pq} := \overrightarrow{pq} \cap \overrightarrow{qp} = \{p + \lambda(q - p) \mid \lambda \in [0, 1]\}$$

with endpoints p and q. The *midpoint* of a segment $\overline{pq}$, where $p = (p_x, p_y)$ and $q = (q_x, q_y)$, is the point $(\frac{1}{2}(p_x + q_x), \frac{1}{2}(p_y + q_y))$.

Define a total order on the elements of $\mathbb{R}^d$, $d \in \mathbb{N}$, as follows. For two elements $p, q \in \mathbb{R}^d$, $d \geq 2$, with $p = (p_1, \ldots, p_d)$ and $q = (q_1, \ldots, q_d)$ define $p \leq q$ if and only if either $p_1 < q_1$ or both $p_1 = q_1$ and $(p_2, \ldots, p_d) \leq (q_2, \ldots, q_d)$. For $d = 1$ we take the usual weak total order on $\mathbb{R}$. This order is called the (weak) *lexicographic order* on $\mathbb{R}^d$. The strict lexicographic order on $\mathbb{R}^d$ is defined analogously.

Distances and Orientations For two points $p = (p_x, p_y)$ and $q = (q_x, q_y)$ denote their (Euclidean) *distance* by

$$\|p - q\| := \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} .$$

Similarly, for a line segment $s = \overline{pq}$ define its *length* as $\|s\| := \|p - q\|$. For a point p and a positive real number ε let

$$\mathcal{B}_\varepsilon(p) := \{q \in \mathbb{R}^2 \mid \|q - p\| < \varepsilon\}$$

denote the (open) *ball* of radius ε around p. Similarly, denote by $\mathcal{S}_\varepsilon(p) := \{q \in \mathbb{R}^2 \mid \|q - p\| = \varepsilon\}$ the *circle* of radius ε around p. For three points $p = (p_x, p_y)$, $q = (q_x, q_y)$, and $r = (r_x, r_y)$ we say that p is to the *left of* the ray $\overrightarrow{qr}$ if and only if

$$\det(p, q, r) := \begin{vmatrix} p_x & p_y & 1 \\ q_x & q_y & 1 \\ r_x & r_y & 1 \end{vmatrix} > 0 \,.$$

Similarly, p is to the *right of* the ray $\overrightarrow{qr}$ if and only if $\det(p, q, r) < 0$, and p, q, and r are *collinear* if and only if $\det(p, q, r) = 0$.

Angles For three points p, q, and r with $p \neq q \neq r$ let $s_i := \mathcal{S}_1(q) \cap \overrightarrow{qi}$ for $i \in \{p, r\}$. The *measured angle* $\measuredangle(p, q, r)$ from p to r around q is defined as the length of the circular arc from s_p to s_r along $\mathcal{S}_1(q)$ in counterclockwise direction. For $s_p = s_r$ or $q \in \{p, r\}$ set $\measuredangle(p, q, r) := 0$. If $\measuredangle(p, q, r) < \pi$ we call $\measuredangle(p, q, r)$ *strictly convex*. If $\measuredangle(p, q, r) > \pi$ we say that $\measuredangle(p, q, r)$ is *reflex*. If $\measuredangle(p, q, r) = \pi$ the angle $\measuredangle(p, q, r)$ is *flat*. An angle that is either strictly convex or flat is referred to as *convex*. The *open angular domain* $\mathcal{D}_{(\measuredangle)}(p, q, r)$ of p and r around q is the set of all points s for which $0 < \measuredangle(p, q, s) < \measuredangle(p, q, r)$. Analogously, the *closed angular domain* $\mathcal{D}_{[\measuredangle]}(p, q, r)$ of p and r around q is the set of all points s for which $\measuredangle(p, q, s) \leq \measuredangle(p, q, r)$.

Obviously it is $\measuredangle(p, q, r) + \measuredangle(r, q, p) = 2\pi$ if and only if p, q, r are pairwise distinct. Sometimes we are just interested in the minimum of these two angles. Let us define the *angle formed by* p and r around q as

$$\sphericalangle(p, q, r) := \min\{\measuredangle(p, q, r), \measuredangle(r, q, p)\} \,.$$

Clearly $\sphericalangle(p, q, r)$ is always convex.

Convex Hull A set $S \subseteq \mathbb{R}^2$ is *convex* if and only if for every two points $p, q \in S$ the line segment $\overline{pq}$ is contained in S. The *convex hull* $\mathrm{conv}(P)$ of a set $P \subseteq \mathbb{R}^2$ is the smallest (with respect to set inclusion) convex superset of P. Since the intersection of an arbitrary family of convex sets is again convex, $\mathrm{conv}(P)$ is well defined and it is the same as the intersection of all convex supersets of P. If P is finite then $\mathrm{conv}(P)$ is an intersection of closed halfplanes that are defined by points from P.

2.4 Topology

A *topological space* is a pair (X, O) where X is a set and $O \subset 2^X$ a set of subsets of X that are called the *open* sets such that

i) $\emptyset$ and X are open,

ii) the union of an arbitrary family of open sets is open,

iii) and the intersection of a finite family of open sets is open.

A set $B \subset O$ is a *basis* of (X, O) if and only if every open set is a union of elements from B. Here we are mostly concerned with $\mathbb{R}^2$ and its metric topology that is generated by the basis $\{\mathcal{B}_\varepsilon(p) \mid p \in \mathbb{R}^2 \text{ and } 0 < \varepsilon \in \mathbb{R}\}$. Still, the term "point" is used within this section to refer to the elements of X as an element of a topological space rather than as an element of $\mathbb{R}^2$.

A set $N \subset X$ is a *neighborhood* of $x \in X$ if and only if there is an open set A such that $x \in A \subseteq N$. In this case x is called an *interior* point of N. The set of interior points of N is denoted by N°. Similarly, for an *exterior* point of N the complement $X \setminus N$ is a neighborhood, and the set of exterior points of N is denoted by $\mathrm{ext}(N)$. The points of X that are neither interior nor exterior to N constitute the *frontier* (or *topological boundary*) ∂N.

A set $A \subset X$ is *closed* if and only if its complement $X \setminus A$ is open. For $B \subset X$ the *closure* of B is denoted by $\overline{B} := B^\circ \cup \partial B$. Clearly both $\overline{B}$ and ∂B are closed. As $\emptyset$ and X are complementary and they are both open, they are also both closed. The space X is said to be *connected* if and only if $\emptyset$ and X are the only subsets of X that are both open and closed. Otherwise X is *disconnected*.

A subset U of a topological space X is *compact* if and only if every open cover of U contains a finite subcover. More precisely, for every family $(O_i)_{i\in I}$ of open subsets of X for which $U \subseteq \bigcup_{i\in I} O_i$, there exists a finite subset $J \subseteq I$ such that $U \subseteq \bigcup_{i\in J} O_i$. A subset of $\mathbb{R}^n$ is compact if and only if it is closed and bounded. Here *bounded* means that the set is contained in a ball of radius k around p, for some $p \in \mathbb{R}^n$ and some $k \in \mathbb{R}$.

Each subset $A \subset X$ together with $O_A := \{B \cap A \mid B \in O\}$ again forms a topological space. The set O_A is referred to as the *subspace*

topology of A in (X, O). Note that the sets in O_A, in particular A, need not be open in X. For two topological spaces (X, O) and (Y, B) the product $X \times Y$ together with the so-called product topology again forms a topological space. The *product topology* of (X, O) and (Y, B) is the topology generated by the basis $\{o \times b \mid o \in O \text{ and } b \in B\}$.

A map $f : X \to Y$ between topological spaces (X, O) and (Y, B) is *continuous* if and only if $f^{-1}(b) \in O$ for all $b \in B$. A *path* between a and b in X is a continuous map $\gamma : [0, 1] \to X$ with $\gamma(0) = a$ and $\gamma(1) = b$. If $a = b$ then γ is a *closed path*. The space X is called *path-connected* if and only if for every two points $a, b \in X$ there is a path between a and b. A path-connected space is always connected but not every connected space is path-connected. But both notions are equivalent for open subsets of $\mathbb{R}^d$. For $X = \mathbb{R}^2$ we will often refer to a path as a *curve* in order to clearly distinguish paths in the topological sense from paths in the graph-theoretic sense.

Two continuous maps $f, g : X \to Y$ between topological spaces X and Y are *homotopic* if and only if there is a continuous map $h : X \times [0, 1] \to Y$ such that $h(x, 0) = f(x)$ and $h(x, 1) = g(x)$, for all $x \in X$. The map h is referred to as a *homotopy* between f and g. A map that is homotopic to a constant map is called *nullhomotopic*. A space is *simply connected* if and only if it is path-connected and every closed path in it is nullhomotopic.

2.5 Geometric Straight Line Graphs

In a *geometric graph*, the vertices are points, and edges are associated to curves connecting their endpoints. We are particularly interested in the case where all of these curves are line segments. But before we can define this class of graphs precisely, we need a more elaborate terminology to describe the interaction between two line segments or, more generally, two subsets of the Euclidean plane.

As a first step, let us define a notion of interior points that is meaningful for non-fulldimensional subsets of the plane, such as line segments.

Definition 2.1 *For a set $P \subset \mathbb{R}^2$ the* **relative interior** *$P^{\ominus}$ is the set of all points $p \in P$ such that*

i) p *is interior to* P

ii) **or** *for every* $\varepsilon > 0$ *there exists* $\delta \leq \varepsilon$ *such that* $\mathcal{B}_\delta(p) \setminus P$ *is not simply connected.*

In particular, for a non-degenerate line segment s it is $s^\ominus = s \setminus V(s)$. If s is degenerate, we have $s^\ominus = s = V(s)$ instead. The following definition provides two refined notions for intersection.

Definition 2.2 *Two sets* $P, Q \subset \mathbb{R}^2$ *are said to* **overlap** *at every point* $c \in P^\ominus \cap Q^\ominus$. *The sets* P *and* Q **cross** *at a point* $c \in P \cap Q$ *if and only if for every* $\varepsilon > 0$ *there exist non-degenerate line segments* s *with* $V(s) \subseteq P \cap \mathcal{B}_\varepsilon(c)$ *and* t *with* $V(t) \subseteq Q \cap \mathcal{B}_\varepsilon(c)$ *such that* $s^\ominus \cap Q \neq \emptyset$ *and* $t^\ominus \cap P \neq \emptyset$.

According to this definition any two sets cross at every common interior point but not necessarily at every point that is in the relative interior of both. However, two non-degenerate line segments overlap at a point c if and only if they cross at c.

Now we have all tools together to define the classes of geometric graphs that we will be concerned with.

Definition 2.3 *A graph* $G = (V, E)$ *with* $V \subset \mathbb{R}^2$ *is a* **geometric straight line graph** (GSLG) *if and only if the segments* $S := \{\overline{uv} \mid \{u, v\} \in E\}$ *do not contain any point from* V *other than their endpoints. In other words, for every segment* $\overline{uv} \in S$ *it is* $V \cap \overline{uv} = \{u, v\}$. *If, moreover, no two segments in* S *cross then* G *is called a* **planar straight line graph** (PSLG). *A graph is* **planar** *if and only if it is isomorphic to a* PSLG.[1]

Obviously, any subgraph of a GSLG (PSLG) is a GSLG (PSLG) as well. Since in a GSLG both an edge and a line segment are uniquely defined by two points, we often identify edges and line segments. In this way any GSLG $G = (V, E)$ induces a set $\partial G := \bigcup_{p \in V \cup E} p$ of points in the plane. Removal of ∂G divides the plane into a finite number of maximal connected open subsets that are called the *faces* of G. The set of faces of G is denoted by $\mathcal{F}(G)$. *Euler's formula* states that

$$|V| - |E| + |\mathcal{F}(G)| - \omega(G) = 1 .$$

[1] The usual definition for planarity allows to connect the embedded vertices by arbitrary Jordan curves. But it is well known that every planar graph has an embedding in which all edges are line segments [84, 34, 89, 83].

As we consider finite PSLGs only it is clear that exactly one of the faces is unbounded. It is sometimes referred to as the *infinite face* of G. In a connected graph every finite face $F \in \mathcal{F}(G)$ is bounded by the edges of a closed trail in G. Such a trail (which is the boundary of a face from $\mathcal{F}(G)$) is called a *facial trail* of G. The infinite face is bounded by the edges of a closed walk in G which is a trail except for the cut-edges which appear twice each. The edges bounding a face are also called *incident* to the face.

A path P in a GSLG G is *simple* if and only if $(V(P), E(P))$ is a PSLG, that is, no two edges of P cross. Similarly we define simple cycles. Obviously every path or cycle in a PSLG is simple.

We will now extend the notion of crossings from subsets of the plane to walks in GSLGs.

A trail $T = (p_1, \ldots, p_n)$ in a GSLG induces a piecewise linear curve γ_T from p_1 to p_n in the plane. While the edges of T do not cross each other by definition, the curve γ_T is not simple, that is, injective in general as the same vertex may appear several times along T. But here our interest is focused at crossings rather than at simplicity. For a walk W in a GSLG let $\partial W := \partial(V(W), E(W)) \subset \mathbb{R}^2$.

Definition 2.4 *Two walks U and W in a* GSLG *cross at a point $p \in \partial U \cap \partial W$ if and only if there exist subwalks U' of U and W' of W for which $|U'|, |W'| \leq 3$ and such that $\partial U'$ and $\partial W'$ cross at p.*

We could require the subwalks in Definition 2.4 to be non-trivial because for a trivial walk U' the set $\partial U'$ consists of a single point. As there no way to build a non-degenerate line segment from a single point set there is no way that any set in the plane can cross a single point.

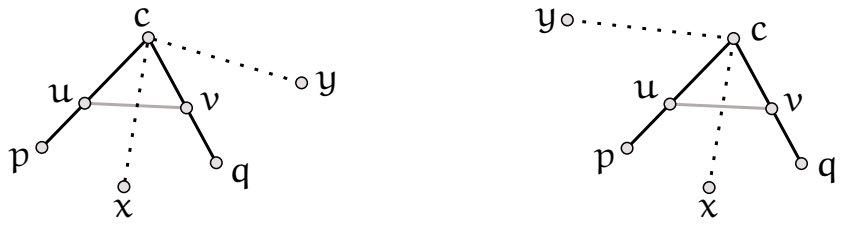
In a PSLG two walks that do not share an edge can cross at vertices only. Let us give an alternative characterization for such crossings: we say that two walks U and W *interleave* at a vertex p of a GSLG $G = (V, E)$ if and only if U contains a subwalk (q, p, r) and W contains a subwalk (s, p, t) such that p, q, r, s, and t are pairwise distinct and $\measuredangle(q, p, r)$ is strictly in between $\measuredangle(q, p, s)$ and $\measuredangle(q, p, t)$. (A number $b \in \mathbb{R}$ is *strictly in between* two numbers $a, c \in \mathbb{R}$ if and only if $a < b < c$ or $c < b < a$.)

Proposition 2.5 *Two walks in a* PSLG *cross if and only if they share an edge or they interleave.*

Proof. "⇒": Consider two walks U and W in a PSLG $G = (V, E)$ that cross at a point $c \in \partial U \cap \partial W$. If $c \notin V$ then by definition of PSLG c lies on a unique edge $e \in E$. The only way c can be in $\partial U \cap \partial W$ is that e appears on both U and W. Otherwise we may assume that U and W cross at vertices only, in particular $c \in V$. Consider the ball $\mathcal{B}_\delta(c)$ with radius $\delta := \frac{1}{2} \min_{p \in \partial(G \setminus c)} \|p - c\|$. Observe that the choice of δ ensures that $V \cap \mathcal{B}_\delta(c) = \{c\}$ and that all edges of G which intersect $\mathcal{B}_\delta(c)$ are incident to c.

By definition there are subwalks U' of U and W' of W with $2 \leq |U'|, |W'| \leq 3$ such that $\partial U'$ and $\partial W'$ cross at c. As U and W cross at vertices only, U' and W' must both be of size three. Since U' and W' cross at c there is a non-degenerate segment $s = \overline{uv}$ with $V(s) \subseteq \partial U' \cap \mathcal{B}_\delta(c)$ such that $s^\ominus \cap \partial W' \neq \emptyset$. If there is an edge $e \in E$ such that $s^\ominus \subset e^\ominus$ then by definition of δ the path U visits e. But on the other hand we have $e^\ominus \cap \partial W' \neq \emptyset$, contrary to our assumption that U and W cross at vertices only. Hence, there is no edge of G containing s. By the choice of δ this implies that u, v, and c are pairwise distinct. Furthermore there must be two distinct edges in U' that are incident to c, in other words, $U' = (p, c, q)$ and, analogously, $W' = (x, c, y)$. If c, p and q are collinear then $\partial U' \subset \overline{pq}$ implies that x and y must lie on different sides of the line through p and q. Otherwise we may assume without loss of generality that q is to the left of $\overrightarrow{cp}$, $u \in \overline{cp}$, and $v \in \overline{cq}$.

Then $s^\ominus \subset \mathcal{D}_{(\measuredangle)}(p, c, q)$ implies that $\mathcal{D}_{(\measuredangle)}(p, c, q) \cap \partial W' \cap \mathcal{B}_\delta(c) \neq \emptyset$. Hence by the choice of δ there must be an edge $f \in E(W')$ with $f^\ominus \subset \mathcal{D}_{(\measuredangle)}(p, c, q)$. Without loss of generality let $f = \overline{cx}$. Consider two cases: If y is left of $\overrightarrow{cx}$ (Figure 8(a)) then by the same argument as above it is $\overline{cq}^\ominus \subset \mathcal{D}_{(\measuredangle)}(x, c, y)$. Otherwise (Figure 8(b)), we have $\overline{cp}^\ominus \subset \mathcal{D}_{(\measuredangle)}(y, c, x)$. In either case U' and W' and, thus, U and W interleave at c.



(a) y is left of $\overrightarrow{cx}$. (b) y is right of $\overrightarrow{cx}$.

Figure 8: *Crossing walks interleave. Edges of one walk are shown by solid lines, edges of the other walk by dotted lines.*

"$\Leftarrow$": Consider two walks U and W in a PSLG $G = (V, E)$. Clearly, if U and W share an edge $e \in E$ they cross each other at every point of $e^{\ominus}$. Hence assume that U and W interleave, and let $U' := (q, p, r)$ be a subwalk of U and $W' := (s, p, t)$ be a subwalk of W such that p, q, r, s, and t are pairwise distinct and $\measuredangle(q, p, r)$ is strictly in between $\measuredangle(q, p, s)$ and $\measuredangle(q, p, t)$. Consider the ball $\mathcal{B}_\varepsilon(p)$ for some arbitrary $\varepsilon > 0$ and the circle $\mathcal{S}_\delta(p) \subset \mathcal{B}_\varepsilon(p)$ where $\delta := \frac{1}{2}\min\{\varepsilon, \min_{p \in V} \|p - c\|\}$. Let $f := \overline{q'r'}$ and $g := \overline{s't'}$ where $i' := \mathcal{S}_\delta(p) \cap \overrightarrow{pi}$, for $i \in \{q, r, s, t\}$. Clearly $V(f) \subseteq \partial U' \cap \mathcal{B}_\varepsilon(p)$ and $V(g) \subseteq \partial W' \cap \mathcal{B}_\varepsilon(p)$. We show that $f^{\ominus} \cap \partial W' \neq \emptyset$. If p, q, and r are collinear then $p \in f^{\ominus} \cap \partial W'$. If $\measuredangle(p, q, r)$ is strictly convex then $\emptyset \neq f^{\ominus} \cap \overrightarrow{ps} \subset \partial W'$. (Note that $\overrightarrow{ps} \cap \mathcal{B}_\delta(p) \subset \partial W'$.) Finally, if $\measuredangle(p, q, r)$ is reflex then $\emptyset \neq f^{\ominus} \cap \overrightarrow{pt} \subset \partial W'$. Analogously one can conclude that $g^{\ominus} \cap \partial U' \neq \emptyset$. Thus U' and W' and therefore U and W cross. □

A walk W in a PSLG is *self-crossing* if and only if it contains two crossing sub-walks, that is, $W = (\ldots, p_i, \ldots, p_j, \ldots, p_k, \ldots, p_\ell, \ldots)$, with $1 \leq i < j \leq k < \ell \leq n$, and the walks $(p_i, \ldots, p_j)$ and $(p_k, \ldots, p_\ell)$ cross. Whenever we want to emphasize that a walk is not self-crossing we express this by referring to it as *non-crossing*. Observe that a walk that is not a trail is always self-crossing.

2.6 Visibility Graphs

For any two points p and q the shortest curve between p and q according to the Euclidean metric is described by the line segment $\overline{pq}$. But the situation changes slightly if we introduce an obstacle region $O \subset \mathbb{R}^2$, say, a line segment and ask for a shortest curve between p and q that does not cross O. If such a curve exists, it is called a *shortest geodesic* curve between p and q with respect to O. This term is used whenever one is interested in shortest curves from a specific class of curves. The curve is called "geodesic" according to a specific instance: points on the surface of a three-dimensional sphere where distance is measured along the surface. Hence, the geodesic stays on the surface, whereas the shortest connection according to the Euclidean metric passes through the interior of the sphere.

Again the term "cross" from above needs some further explanation. It turns out that the notion of crossing we want to use for defining

geodesics is different from the one that has been discussed in Section 2.5. In order to avoid confusion of both concepts let us assign a completely different name to this new concept: *penetration*. Intuitively we want to allow geodesics to walk along ∂O (which implies a crossing) but not to pass through to "the other side". From now on we will only consider obstacle sets that consist of (compact regions that are bounded by) a finite number of line segments. Let us refer to these sets as *finite linearly bounded* obstacle sets.

Definition 2.6 *Consider a (rectifiable) curve $\gamma : [0,1] \to \mathbb{R}^2$ and a finite linearly bounded obstacle set $O \subset \mathbb{R}^2$.*

The set O is said to be **locally to the right** *of γ at $x \in (0,1)$ if and only if for every $\varepsilon > 0$ there exists some $\delta > 0$ such that for all pairs $x^-, x^+ \in (0,1)$ for which $x - \delta < x^- < x < x^+ < x + \delta$ it is*

$$\mathcal{D}_{(\measuredangle)}(\gamma(x^-), \gamma(x), \gamma(x^+)) \cap \mathcal{B}_\varepsilon(\gamma(x)) \cap O \neq \emptyset \,.$$

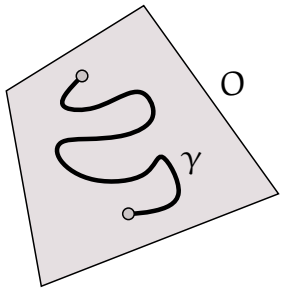
Similarly the set O is **locally to the left** *of γ at $x \in (0,1)$ if and only if for every $\varepsilon > 0$ there exists some $\delta > 0$ such that for all pairs $x^-, x^+ \in (0,1)$ for which $x - \delta < x^- < x < x^+ < x + \delta$ it is*

$$\mathcal{D}_{(\measuredangle)}(\gamma(x^+), \gamma(x), \gamma(x^-)) \cap \mathcal{B}_\varepsilon(\gamma(x)) \cap O \neq \emptyset \,.$$

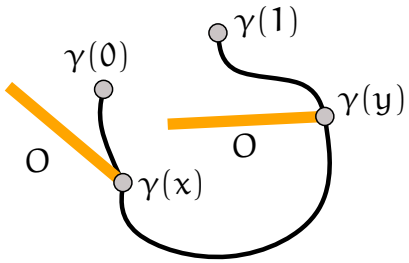
We say that γ **penetrates** *O at an interval $[x,y] \subset (0,1)$ (possibly $x = y$) if and only if $\gamma([x,y]) \cap O^\circ \neq \emptyset$ or both $\gamma([x,y]) \subseteq \partial O$ and O is locally on different sides of γ at x and y.*

Figure 9 shows a few examples illustrating Definition 2.6. In general penetration cannot be observed locally. This is in contrast to the notions of overlap and crossings defined in Section 2.5 that are both tied to the neighborhood of a single point. Also note that penetrations are independent of the direction of the curve. In other words, if γ penetrates a set O then also $\gamma' := (x \mapsto \gamma(1-x))$, for $x \in [0,1]$, penetrates O. Hence we may define that a segment s *penetrates* a set O if and only if the curve that traverses s uniformly penetrates O. Then two segments penetrate if and only if they cross and are not collinear. In particular no segment penetrates itself.

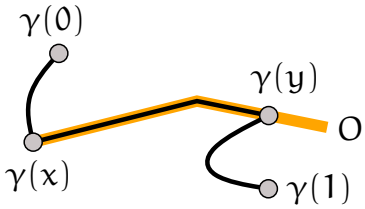
In the same way as for segments we define that a walk W in a GSLG *penetrates* a set O if and only if the curve γ_W that traverses W



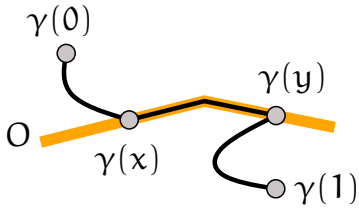
(a) γ penetrates O because $\gamma([0,1]) \cap O^\circ \neq \emptyset$.



(b) γ does not penetrate O because $\gamma([x,y]) \not\subseteq \partial O$.



(c) γ does not penetrate O because O is nowhere locally to the right of γ.



(d) γ penetrates O because $\gamma([x,y]) \subseteq \partial O$ and γ is locally on different sides at x and at y.

Figure 9: *Examples for (non-)penetration.*

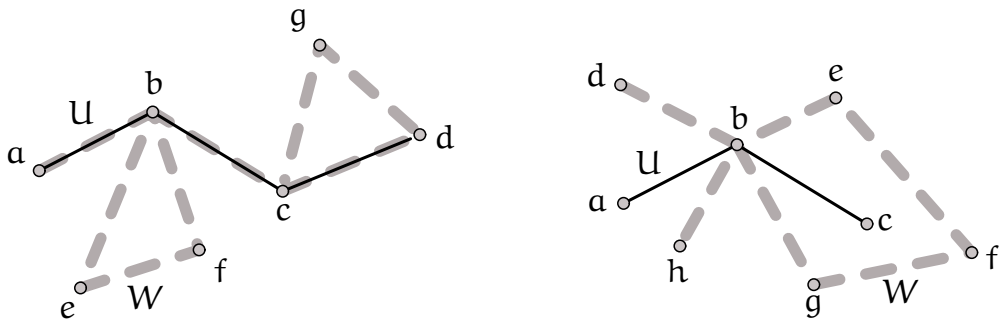
uniformly penetrates O. If γ_W penetrates O at an interval $[x,y] \subset (0,1)$, we say that W *penetrates* O *at* $\gamma([x,y])$.

For a walk W in a GSLG the set ∂W is finite linearly bounded. Thus we can define that for two walks U and W in a GSLG U *penetrates* W if and only if U crosses ∂W. Unless in the case of line segments this penetration relation is in general not symmetric as the examples in Figure 10 demonstrate, not even if U and W are edge-disjoint (Figure 10(b)). Consequently a walk W is called *self-penetrating* if and only if W penetrates ∂W.

If we think of the obstacle as blocking sight then two points whose connecting line segment crosses O cannot "see each other". This interpretation gives rise to the notion of visibility graphs.

Definition 2.7 *Consider a set* P *of* n *points and a set* $O \subset \mathbb{R}^2$. *The* **visibility graph** *of* P *with respect to* O *is the* GSLG $G = (P, E)$ *where* E *consists of all edges* $\{p,q\} \in \binom{P}{2}$ *for which the segment* $s := \overline{pq}$ *does not penetrate* O *and* $s^{\ominus} \cap P = \emptyset$.

Observe that a visibility graph is indeed a GSLG but in general not a



(a) $U = (a, b, c, d)$ **penetrates** $W = (a, b, e, f, b, c, d, g, c)$ **at** $\overline{bc}$.

(b) $U = (a, b, c)$ **penetrates** $W = (d, b, e, f, g, b, h)$ **at** b.

Figure 10: *Two walks* U *and* W *such that* U *penetrates* ∂W *but* W *does not penetrate* ∂U.

PSLG: For points in general position (no three collinear) and $O = \emptyset$ the visibility graph is isomorphic to K_n and hence non-planar for $n > 4$.

One particular visibility graph we will be concerned with in some detail is the *segment endpoint visibility graph* $\mathrm{Vis}(S)$ that is defined on a set S of n pairwise disjoint line segments, $n \in \mathbb{N}$. It is the visibility graph on the segment endpoints $V(S) := \bigcup_{s \in S} V(s)$ with respect to the segments, that is, $\partial S := \bigcup_{s \in S} s$. If the segments in S are non-degenerate the graph $\mathrm{Vis}(S)$ has $2n$ vertices and two types of edges: those which correspond to segments from S (recall that a segment does not penetrate itself) are called *segment edges*, the remaining edges are referred to as *visibility edges*. See Figure 11 for an example.

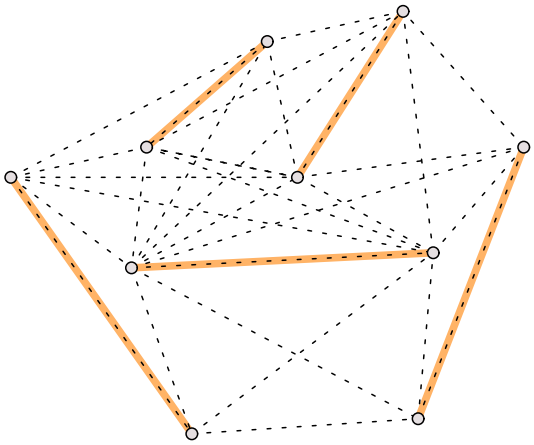


Figure 11: *A segment endpoint visibility graph. The segments of* S *are shown by thick solid lines and the edges of* $\mathrm{Vis}(S)$ *by dotted lines.*

An equivalent definition of $\mathrm{Vis}(S)$ would be to say that two points $p, q \in V(S)$ are connected if and only if either $\overline{pq} \in S$ or $\overline{pq}^{\ominus} \cap \partial S = \emptyset$. Also note that $\mathrm{Vis}(S)$ is different from the so-called *segment visibility graph* (or *full visibility graph*), where vertices correspond to segments and an edge connects two vertices if and only if some points of the two segments are mutually "visible".

Our goal in the following is to establish a connection between visibility graphs and geodesics. Indeed, the following proposition tells us that the visibility graph contains all shortest paths.

Proposition 2.8 *Consider a set P of points and a finite linearly bounded obstacle set O such that $P \cap O^\circ = \emptyset$. Denote the set of segments bounding O by $E(O)$ and the corresponding set of segment endpoints by $V(O)$. Then for any pair $p, q \in P$ every shortest curve between p and q that does not penetrate O is a simple path in the visibility graph of $P \cup V(O)$ with respect to O.*

Proof. Clearly the visibility graph of $P \cup V(O)$ with respect to O contains the shortest curve between every pair of points for which the shortest curve (that does not penetrate O) is actually a segment.

Now consider two points $p, q \in P$ for which $\overline{pq}$ penetrates O and let γ be a shortest curve from p to q that does not penetrate O. Consider an $x \in [0, 1]$ for which $\gamma(x) \notin V(O)$.

Either $\gamma(x)$ is in the relative interior of a segment $s \in E(O)$: then there is a neighborhood N of $\gamma(x)$ for which $N \cap V(O) = \emptyset$. As γ is continuous and does not penetrate O there exists a neighborhood M of x such that $\gamma(M) \subset N$ and $\gamma(M)$ is contained in a closed halfplane H through s. Consider any pair $x^-, x^+ \in M$ with $x^- < x < x^+$. Observe that $\gamma([x^-, x^+]) \in H \cap N$ and that no curve in $H \cap N$ penetrates O. From the optimality of γ we can conclude that $\gamma([x^-, x^+])$ is a line segment.

Or there is a neighborhood N of $\gamma(x)$ that is disjoint from O and we can argue as above.

In summary we have shown that γ is locally a segment almost everywhere, that is, except for at the points of $V(O)$. Therefore γ induces a walk in the visibility graph of $P \cup V(O)$ with respect to O. As no walk

in the visibility graph penetrates O we conclude by the optimality of γ that this walk must be a simple path. □

Note that we did not even define the length of a curve in $\mathbb{R}^2$. All we used in the proof of Proposition 2.8 is that in the standard setting without any obstacles the shortest curve connecting two points is a line segment. Curves that correspond to a path in a PSLG are also called finite *simple polygonal* curves. In order to formally define geodesics we need the notion of homotopy for curves with respect to obstacles. According to Proposition 2.8 we may restrict our attention to finite simple polygonal curves.

Definition 2.9 *Two curves α and β are* **homotopic** *with respect to a set O of obstacles if and only if there exists a* **homotopy** $h : [0,1]^2 \to \mathbb{R}^2$ *such that*

i) $h(x,0) = \alpha(x)$ *and* $h(x,1) = \beta(x)$*, for all* $x \in [0,1]$*,*

ii) $h(x,y_0)$ *is a finite simple polygonal curve that does not penetrate O, for all* $y_0 \in [0,1]$*, and*

iii) $h(0,y) = \alpha(0) = \beta(0)$ *and* $h(1,y) = \alpha(1) = \beta(1)$*, for all* $y \in [0,1]$*.*

There is one more ingredient we need in order to finally provide the definition for geodesics. Denote the *length* of a walk $W = (v_1, \ldots, v_k)$, $k \in \mathbb{N}$, in a GSLG by

$$\text{length}(W) := \sum_{i=1}^{k-1} \|v_i - v_{i+1}\| .$$

The length of a walk should not be confused with its size, that is, its number of vertices.

Definition 2.10 *Let $G = (P \cup V(O), E(O))$ be a visibility graph on a set P of n points with respect to a finite linearly bounded obstacle set O. Consider a walk $W = (v_1, \ldots, v_k)$ in G that does not penetrate O. A* **geodesic** $\text{geo}(W)$ *of W in G is a walk in G that has minimum length among all walks in G that are homotopic to W with respect to O.*

Even if W is a simple path the geodesic of W need not be simple as the example in Figure 12 shows. But if W forms the boundary of a convex polygon (polygons are defined and discussed in the next section), in particular a triangle, then $\text{geo}(W)$ is always simple.

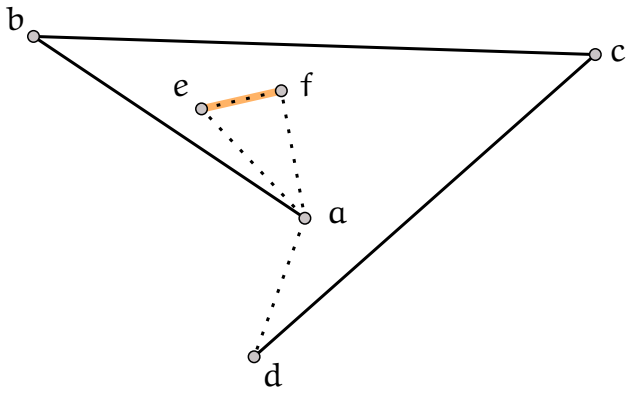


Figure 12: *The geodesic of the path* $P = (a, b, c, d)$ *with respect to the line segment* $\overline{ef}$ *is not simple:* $\text{geo}(P) = (a, e, f, a, d)$.

It is not hard to see that the geodesic is uniquely defined by the walk: suppose for some walk there are two geodesics α and β. As α and β are homotopic with respect to O, the region between any two successive common points of them is a simple polygon whose interior is disjoint from O. But then α and β coincide because shortest paths inside a simple polygon are unique.

2.7 Polygons

The goal of this section is to provide a rigorous definition for polygons. One important property of polygons is that they have a well-defined oriented boundary that corresponds to a cyclic ordering of its edges. In the terminology of geometric graphs the latter corresponds to an Euler tour. We show below that any non-crossing Euler tour provides a consistent orientation for each face of the graph.

Proposition 2.11 *Let* T *be a non-crossing Euler tour of a* PSLG $G = (V, E)$*, and let* C *be any facial trail of* G*. Then* T *visits the edges of* C *in their order along* C*, that is, if each edge of* C *is oriented in the direction it is traversed by* T *then the result is a directed non-crossing trail.*

Proof. Denote $H := G[V(C)]$ and let D be the orientation of H induced by T as described above. Consider a vertex v of H and the cyclic order of the vertices in $N_G(v)$ around v. Let u and x be two vertices of H such that u, v, and x are adjacent along C and hence in this order around v. Suppose that both edges (u, v) and (x, v) in D are oriented towards v. Let $w \in N_G(v)$ be the vertex visited by T directly after (u, v) and let $y \in N_G(v)$ be the vertex visited by T directly after (x, v). Obviously u, v, w, x, and y are pairwise distinct.

As T is non-crossing, by Proposition 2.5 u and w as well as x and y are neighbors in the cyclic order around v. Thus the counterclockwise order around v is without loss of generality u, w, y, x, as depicted in Figure 13. Now recall that $T = (\ldots, u, v, w, \ldots, x, v, y, \ldots)$. In partic-

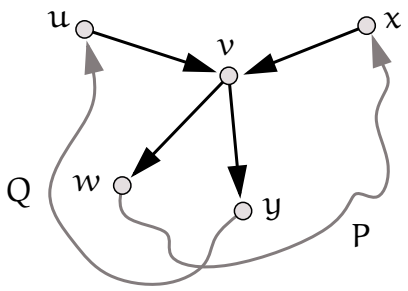


Figure 13: *There can be no vertex v with* $\deg^-_D(v) = 2$.

ular T contains a closed subtrail P from v via w to x and back to v. Similarly T contains a closed subtrail Q from v via y to u and back to v. Moreover P and Q form a partition of T, in particular they are edge-disjoint. Also note that P and Q cross at v and that except for (x, v, w) on P and (u, v, y) on Q all other size three subtrails of P and Q also appear in T. Hence any crossing of P and Q other than at v would also be a crossing of T, in contradiction to our assumption that T is non-crossing. We will show such a crossing exists and thus there can be no vertex v with $\deg^-_D(v) = 2$.

First we may assume that $\deg_P(v) = 2$: as T is non-crossing, there is no subtrail of size three in P that interleaves with one of (x, v, y) or (u, v, w). Moreover u, v, and x are adjacent along a facial cycle and hence there is no edge incident to v in $\mathcal{D}_{(\measuredangle)}(x, v, u)$. Combining these statements we conclude that no subtrail of P other than (x, v, w) interleaves with (u, v, y). Consider the subtrail P_1 of P that ends at the first visit of v by P within $\mathcal{D}_{(\measuredangle)}(y, v, u)$. Such a visit exists because $P = (\ldots, x, v)$ and $x \in \mathcal{D}_{(\measuredangle)}(y, v, u)$. Now let P_2 be the subtrail of

P_1 that starts with the last visit of v by P within $\mathcal{D}_{(\measuredangle)}(u, v, y)$. Again such a visit exists because $P_1 = (v, w, \ldots)$ and $w \in \mathcal{D}_{(\measuredangle)}(u, v, y)$. Now P_2 and Q are two closed edge-disjoint trails that cross at exactly one vertex, namely v, and $\deg_P(v) = 2$.

Next we show that it is impossible that two closed edge-disjoint trails cross at exactly one vertex. Construct a cycle P' from P by traversing P and leaving each vertex according to its last occurrence along P. Clearly $P' = (v, w, \ldots, x, v)$. As P' is a cycle in a PSLG the curve $\gamma_{P'}$ that traverses P' uniformly is a closed Jordan curve. By the Jordan-Curve Theorem $\gamma_{P'}$ divides the plane into a bounded and an unbounded component. As P' crosses (u, v, y) exactly once, at v, so does $\gamma_{P'}$ and thus u and y are separated by $\gamma_{P'}$. In particular there is a second crossing between $\gamma_{P'}$ and Q apart from the crossing at v which corresponds to a second crossing of P' and Q. As P' and Q are edge-disjoint, this crossing occurs at a vertex $z \neq v$.

Either this crossing of P' and Q at z corresponds to a crossing of P and Q and we are done. Otherwise it is $P' = (v, \ldots, a, z, b, \ldots)$ and $P = (v, \ldots, a, z, c, \ldots, d, z, b, \ldots)$. In this case we have found a closed subtrail $P'' := (z, c, \ldots, d, z)$ of P and thus of T such that P'' and Q cross at exactly one vertex: z. Clearly $|P''| < |P|$ and we conclude by induction on $|P|$ that there exists a second crossing between P and Q which implies a self-crossing of T. (Regarding the base case for the induction note that a cycle cannot be crossed by an edge-disjoint closed trail in exactly one vertex as a consequence of the Jordan-Curve Theorem.) □

Having established this nice property of non-crossing Euler tours, we want to ensure that such a tour always exists, as long as the graph under consideration is Eulerian at all.

Theorem 2.12 *Let $G = (V, E)$ be an Eulerian* PSLG*. Then there is an Euler tour T in G such that T is non-crossing.*

Proof. Let $T = (p_1, \ldots, p_n)$ be an arbitrary Euler tour of G. By definition of Euler tour T cannot cross itself at points in the relative interior of its edges. Assume T has a self-crossing at a vertex $p_i = p_j$, that is,

$$T = (p_1, \ldots, p_i, p_{i+1}, \ldots, p_{j-1}, p_i, p_{j+1} \ldots, p_n) ,$$

for some $2 < i+1 < j < n$. Construct another Euler tour T' from T by traversing the sub-path $(p_{i+1}, \ldots, p_{j-1})$ in reversed order. That is,

$$T' = (p_1, \ldots, p_i, p_{j-1}, \ldots, p_{i+1}, p_i, p_{j+1}, \ldots, p_n) .$$

Consider the sorted (increasingly) vector $\alpha(T)$ of angles formed by all triples of points that are consecutive along T (including $p_{n-1}p_np_1$ and $p_np_1p_2$). The tours T and T' differ in exactly two angles:

$$\alpha := \sphericalangle(p_{i-1}, p_i, p_{i+1}) \text{ and } \beta := \sphericalangle(p_{j-1}, p_i, p_{j+1}) \text{ in } T \text{ versus}$$

$$\gamma := \sphericalangle(p_{i-1}, p_i, p_{j-1}) \text{ and } \delta := \sphericalangle(p_{i+1}, p_i, p_{j+1}) \text{ in } T'.$$

We claim that $\alpha(T') < \alpha(T)$ in the lexicographic ordering. Then, whenever there is a self-crossing in T, we can use the above operation to construct another Euler tour T' of G whose vector of angles is strictly smaller than the one of T. Since there are only finitely many distinct Euler tours for G, there must be one that is non-crossing.

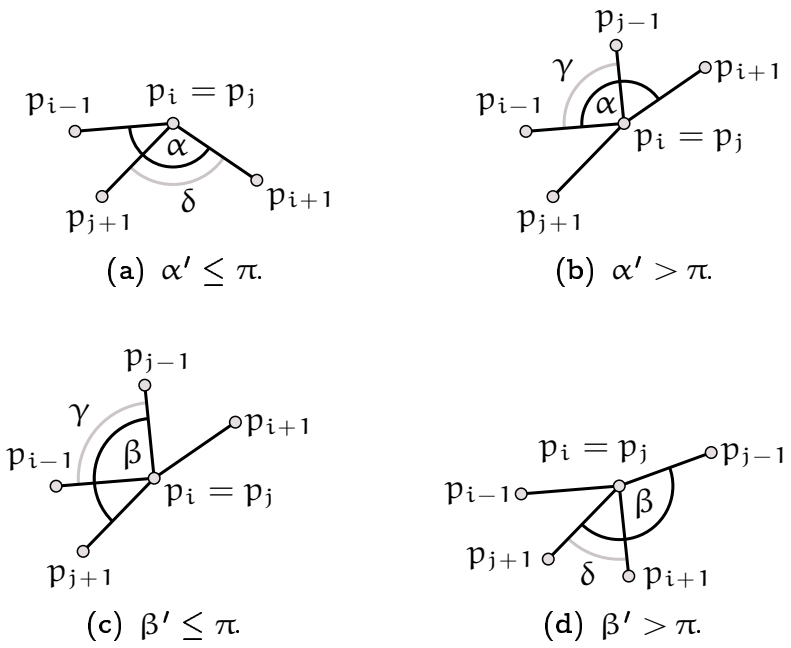
It remains to prove the claim, that is, $\min\{\gamma, \delta\} < \min\{\alpha, \beta\}$. As T has a self-crossing at p_i we may assume by Proposition 2.5 that $\alpha' := \measuredangle(p_{i-1}, p_i, p_{i+1})$ is strictly in between $\gamma' := \measuredangle(p_{i-1}, p_i, p_{j-1})$ and $\measuredangle(p_{i-1}, p_i, p_{j+1})$. That is, the cyclic order of p_{i-1}, p_{i+1}, p_{j-1}, and p_{j+1} around p_i is either $p_{i-1}, p_{j-1}, p_{i+1}, p_{j+1}$ or $p_{i-1}, p_{j+1}, p_{i+1}, p_{j-1}$. Clearly, the minimum angle μ formed by any two of these points around p_i is formed by two neighbors, that is, points that are consecutive in one of these cyclic orders. (In fact, the neighbors are the same in both orders.) As T is an Euler tour, the vertices p_{i-1}, p_{j-1}, p_{i+1}, and p_{j+1} are pairwise distinct. In particular, p_{i-1} and p_{i+1} as well as p_{j-1} and p_{j+1} are not neighbors in the cyclic order around p_i. Thus $\mu < \min\{\alpha, \beta\}$.

If $\mu \in \{\gamma, \delta\}$ then the claim is immediate. Otherwise, assume without loss of generality that $\mu = \sphericalangle(p_{i-1}, p_i, p_{j+1})$. Clearly it is $\mu < \pi$. We distinguish two cases.

Case 1: $\mu' := \measuredangle(p_{i-1}, p_i, p_{j+1}) < \pi$. Consider the location of p_{i+1}: If $\alpha' \leq \pi$ then $\delta < \alpha$ (Figure 14(a)). Else we have $\alpha' > \pi$ and $\gamma < \alpha$ (Figure 14(b)). Similarly, if $\beta' := \measuredangle(p_{j-1}, p_i, p_{j+1}) \leq \pi$ then $\gamma < \beta$ (Figure 14(c)), and for $\beta' > \pi$ it is $\delta < \beta$ (Figure 14(d)).

Case 2: $\mu' := \measuredangle(p_{i-1}, p_i, p_{j+1}) > \pi$. Consider the location of p_{i+1}: If $\alpha' \leq \pi$ then $\gamma < \alpha$. Else we have $\alpha' > \pi$ and $\delta < \alpha$. Similarly, if $\beta' \leq \pi$ then $\delta < \beta$, and for $\beta' > \pi$ it is $\gamma < \beta$.

This completes the proof of both the claim and the theorem. □



(a) $\alpha' \leq \pi$. (b) $\alpha' > \pi$.

(c) $\beta' \leq \pi$. (d) $\beta' > \pi$.

Figure 14: *Angles around a self-crossing of the tour in Case 1 ($\measuredangle(p_{i-1}, p_i, p_{j+1}) < \pi$). The angles of T are shown by black arcs, the angles of T' by grey arcs.*

Theorem 2.12 asserts the existence of a non-crossing Euler tour in every Eulerian PSLG. Of course, a specific PSLG might contain several such tours. Are there graphs which have exactly one non-crossing Euler tour? If a tour is considered as a sequence of vertices then the only such graphs are those with only one vertex. This is because for every Euler tour T of a graph G traversing the vertices in reverse order yields another Euler tour T' of G which is distinct from T if it consists of more than one vertex. Other tours can be generated by shifting the sequence of vertices cyclically.

Let us exclude these generally applicable options by declaring two closed walks $U = (u_0, \ldots, u_k)$ and $W = (w_0, \ldots, w_k)$, for some $k \in \mathbb{N}$, in a graph G *equivalent* if and only if there is some j with $0 \leq j \leq k$ such that $w_i = u_{\varphi(i)}$, for all $0 \leq i \leq k$, where either $\varphi(i) = (i + j) \bmod (k + 1)$ or $\varphi(i) = k - ((i + j) \bmod (k + 1))$.

We will define polygons to be PSLGs that have a certain type of Euler tour. But we want to allow polygons to degenerate into line segments locally, that is, in the terminology of PSLGs we want to allow cut-edges. Clearly a graph that has a cut-edge is not Eulerian. Hence we allow

the tour to traverse each cut-edge twice. Using the following simple construction we can consider polygons to be Eulerian graphs for most purposes although strictly speaking they are not neccesarily Eulerian.

Definition 2.13 *For a* PSLG $G = (V, E)$ *its* **2-edge-connected closure** $G_{\overline{2}} = (V', E')$ *is defined as follows. Denote by* $C \subseteq E$ *the set of cut-edges of* G.

The vertex set of $G_{\overline{2}}$ *is* $V' := V \cup \{v_c \mid c \in C\}$ *where* v_c *is constructed such that its distance to the midpoint* m_c *of* c *is* δ *and* v_c *is to the right of* $\overrightarrow{xy}$*, where* x *is the lexicographically smaller endpoint of* c *and* $\delta := \frac{1}{2}\min_{p \in \partial(G \setminus c)} \|p - m_c\|$.

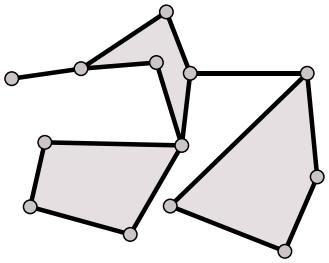
The edge set of $G_{\overline{2}}$ *is* $E' := E \cup \{\{x, v_c\} \mid c \in C \text{ and } x \in V(c)\}$.

Observe that $G_{\overline{2}}$ is always a PSLG. An Euler tour in $G_{\overline{2}}$ induces a tour of G where every subpath of the type (x, v_c, y) for some cut-edge $c = \overline{xy}$ of G is replaced by the path (x, y). By definition of $G_{\overline{2}}$ the corresponding tour of G is a trail apart from the fact that cut-edges are traversed twice.

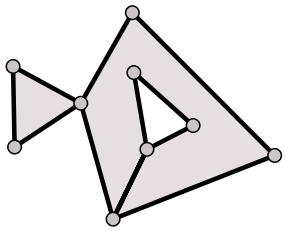
Definition 2.14 *A* PSLG $P = (V, E)$ *is a* **polygon** *if and only if all cut-edges of* P *are incident to its infinite face and* $P_{\overline{2}}$ *contains a unique (up to equivalence) non-crossing Euler tour. The corresponding tour of* P *is denoted by* $P_{\circlearrowleft}$. *The set* $\partial P \subset \mathbb{R}^2$ *is called the* **boundary** *of* P.

The polygon P *is* **simple** *if and only if* $P_{\circlearrowleft}$ *is a cycle. A simple polygon on three vertices is a* **triangle**, *a simple polygon on four vertices is referred to as a* **quadrilateral**.

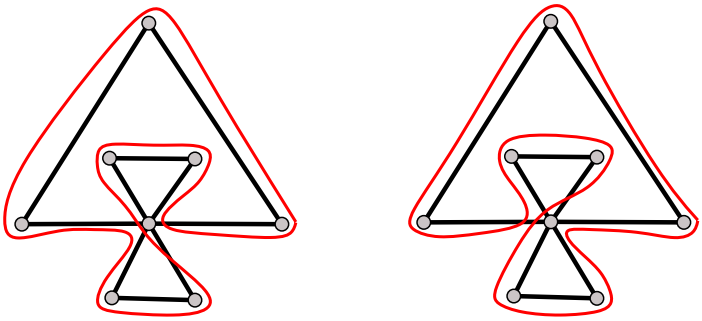
Note that we allow *degenerate* polygons that consist of a single point or line segment only. As the symbol indicates we always assume that $P_{\circlearrowleft}$ visits the vertices of the infinite face of P in counterclockwise order. For $P_{\circlearrowleft} = (p_0, \ldots, p_k)$ with $p_0 = p_k$ and a vertex p_i, for $0 \leq i \leq k$, denote by $p_{i\oplus 1} := p_{(i+1) \bmod k}$ the *successor* of p_i and by $p_{i\ominus 1} := p_{(i+k-1) \bmod k}$ the *predecessor* of p_i in $P_{\circlearrowleft}$. We say that p_i, for $0 \leq i < k$, is a *convex vertex* in $P_{\circlearrowleft}$ if and only if $\measuredangle_P p_i := \measuredangle(p_{i\oplus 1}, p_i, p_{i\ominus 1})$ is convex. Analogously define *strictly convex*, *reflex*, and *flat vertices* of $P_{\circlearrowleft}$. Observe that being a (strictly) convex, reflex, or flat vertex is not a property of the point p_i alone which might appear several times in $P_{\circlearrowleft}$. Instead this property is tied to a specific occurrence of p_i at position i in $P_{\circlearrowleft}$.



(a) A polygon.



(b) Not a polygon: a cut-edge is not incident to the infinite face.



(c) Not a polygon: two distinct non-crossing Euler tours.

Figure 15: *Examples of (non)-polygons.*

By Proposition 2.11 every face of $P_{\overline{2}}$ is bounded by a directed non-crossing trail in the orientation of $P_{\overline{2}}$ induced by the Euler tour. If this directed trail is oriented counterclockwise, we call the corresponding face *positively oriented*. As each face of P directly corresponds to a face of $P_{\overline{2}}$, we also get an orientation for the faces of P in this way. The tour $P_{\circlearrowleft}$ of P traverses the boundary of each face consistently if we allow cut-edges to be traversed once in both directions.

Denote the set of positively oriented finite faces of P by $\mathcal{F}^+(P)$. To every polygon P we associate the *polygonal domain* $\mathcal{R}(P)$ that is defined as

$$\mathcal{R}(P) := \partial P \cup \bigcup_{F \in \mathcal{F}^+(P)} F .$$

The *interior* of a polygon P is denoted by $P^\circ := \mathcal{R}(P) \setminus \partial P$. Similarly the *exterior* of P is defined as $\mathrm{ext}(P) := \mathbb{R}^2 \setminus \mathcal{R}(P)$. The following proposition justifies the naming of these regions. Also observe the implication that $\mathcal{R}(P)$ is compact.

Proposition 2.15 *∂P is the frontier (topological boundary) of $\mathcal{R}(P)$.*

Proof. Recall that the frontier of a set $M \subset \mathbb{R}^2$ consists of all points in $\mathbb{R}^2$ that are neither interior to M nor to its complement $\mathbb{R}^2 \setminus M$. Denote the frontier of P by $B(P)$. Clearly it is $B(P) \subset \partial P$ because all faces are open and hence all their points are interior. On the other hand consider a point $p \in \partial P$.

If p lies in the relative interior of some edge $e \in E(P)$ then consider the orientation of e induced by $P_{\circlearrowleft}$. If e is a cut-edge of P then by definition of polygon e is incident to the infinite face of P which is exterior to P. Otherwise, by definition of $\mathcal{R}(P)$ the face to the left of e belongs to P° while the face to the right of e belongs to $\mathrm{ext}(P)$. Thus every neighborhood of p intersects both $\mathcal{R}(P)$ and $\mathrm{ext}(P)$.

If on the other hand $v \in V(P)$ then there is at least one edge of P incident to v because $P_{\overline{2}}$ is Eulerian. In particular every open neighborhood N of p contains a point q in the relative interior of some edge. As N is also a neighborhood of q we can argue as above. □

Sometimes it is convenient to subdivide a large polygon into several smaller polygons each of which is potentielly easier to handle than the original large polygon. Ideally such a subdivision is not too redundant in covering some parts of the polygon many times.

Definition 2.16 *For a polygon P a* **dissection** *is a finite set $\{P_1, \ldots P_n\}$ of polygons such that $\bigcup_{1 \le i \le n} \mathcal{R}(P_i) = \mathcal{R}(P)$ and $\mathcal{R}(P_i)$ does not overlap $\mathcal{R}(P_j)$, for every $1 \le i, j \le n$ where $i \neq j$.*

Apart from simple polygons there are a number of other important classes of polygons. Within this work we will be concerned with simply connected polygons most of the time.

Definition 2.17 *A polygon P is* **simply connected** *if and only if $\mathcal{R}(P)$ is simply connected. A polygon P is* **convex** *if and only if $\mathcal{R}(P)$ is convex.*

The convex hull of a finite set whose points are not all collinear is a convex polygon.

Chapter 3

Hamiltonian Polygons

This chapter is devoted to the study of Hamiltonian polygons in segment endpoint visibility graphs.

Definition 3.1 *For a set S of pairwise disjoint line segments, a* **Hamiltonian polygon** *is a simple polygon that is a spanning subgraph of* $\mathrm{Vis}(S)$.

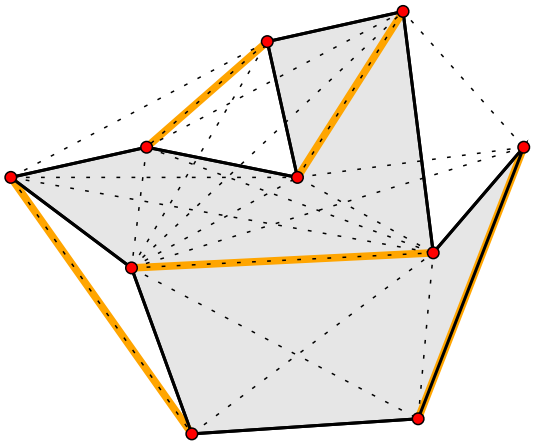


Figure 16: *A Hamiltonian polygon H for a set S of segments. The segments from S are shown by thick solid gray lines, the polygon H by thin solid black lines, and the edges of* $\mathrm{Vis}(S)$ *by dotted lines.*

The segments from S can be grouped into three different categories according to their position relative to a Hamiltonian polygon H: some of

them may appear as edges of H, others are *diagonals*, that is, segments whose relative interior is contained in $\mathcal{R}(H)$, and the third kind is called *epigonals*, that are segments whose relative interior is disjoint from $\mathcal{R}(H)$.

Obviously the existence of a Hamiltonian polygon implies that the corresponding visibility graph is Hamiltonian. Therefore the main theorem of this chapter – stated below – implies that for (almost) any finite set of pairwise disjoint line segments the endpoint visibility graph is Hamiltonian.

Theorem 3.2 *For any finite set of pairwise disjoint line segments, not all collinear, there exists a Hamiltonian polygon.*

The proof of Theorem 3.2 is algorithmic, that is, for the given set S of segments we construct a Hamiltonian polygon. To simplify the discussion let us assume that all line segments are non-degenerate, that is, all segment endpoints are pairwise distinct. The extension to degenerate segments is straightforward and will be discussed at the end of this chapter.

The next section provides an overview and describes the algorithm and its different phases from a high level point of view.

3.1 Algorithmic Overview

The general idea behind our algorithm is to maintain a polygon within the visibility graph that contains all segment endpoints (in its polygonal domain, not necessarily as vertices) and whose vertices are (some of the) segment endpoints. The convex hull $\mathrm{conv}(\partial S)$ serves as a starting point: it is certainly a subgraph of $\mathrm{Vis}(S)$, it contains all segments, and its vertices are segment endpoints. During the construction we aim at increasing the number of segment endpoints that the current polygon visits by modifying the polygon locally according to certain rules.

At the same time we maintain a dissection of the polygon that tries to nicely group the segments that are still in the interior of the polygon. Here "nicely" means that every segment that is in the interior of the polygon is contained in exactly one polygon of the dissection. Our goal is two-fold: on one hand to include more and more segment endpoints

as vertices into the polygon and on the other hand to simplify the dissection such that eventually all polygons in the dissection are convex.

The nice property of convex dissection polygons is that they put us into exactly the situation where we started: a convex set—initially the convex hull, now the dissection polygon—that possibly contains some segments from S in its interior. At this point we can apply induction to solve the problem for each of the dissection polygons separately. Some care has to be taken that these recursively obtained Hamiltonian polygons for the subproblems can be connected to the "global" polygon to finally form a Hamiltonian polygon for the whole set of segments.

Let us gradually add more detail to the description of the algorithm until we finally get to a fully fledged list of invariants. The construction proceeds in six phases that are illustrated in Figure 17.

Phase 1 Initialisation: Start with the polygon $P \leftarrow \mathrm{conv}(\partial S)$ as shown in Figure 17(a).

Phase 2 Saturation: For those segments for which only one endpoint appears as a vertex of P, include the other endpoint as well using a local operation (Figure 17(b)). As a result of the local modifications the polygon P may visit some segments that have been interior to it originally. In the course of these modifications P will in general become a non-simple polygon.

Phase 3 Dissection: The interior of the polygon P is dissected into convex polygons by cutting it along rays starting from its reflex vertices. Whenever such a ray hits a segment that lies in the interior of P, this segment is integrated into P again by means of a local modification. At the end of this phase all segments from S are either edges of P, diagonals of P, epigonals of P, or they lie in the interior of one of the dissection polygons (Figure 17(c)).

Phase 4 Simplification: Using once again local modifications the polygon P is changed such that it becomes a simple polygon (Figure 17(d)). For this to work we have to carefully control the type of "non-simplicities" that arise in Phase 2 and 3.

Phase 5 Induction: For every convex polygon C in the dissection of our polygon P compute inductively a Hamiltonian polygon that visits the endpoints of all segments that lie in the interior of C (Figure 17(e)). It might happen that there is no segment or just one

segment in the interior of such a polygon which just simplifies the computation in this phase.

Phase 6 Bridging: The Hamiltonian polygons computed in Phase 5 for the segments interior to the dissection polygons are combined with the polygon P to form a Hamiltonian polygon for the whole set S (Figure 17(f)). To ensure that this bridging is always possible we make use of some freedom regarding the construction of P: we can fix an arbitrary edge of the convex hull such that it is part of the Hamiltonian polygon to be constructed. On the other hand we have to make sure that each dissection polygon has a common edge with P.

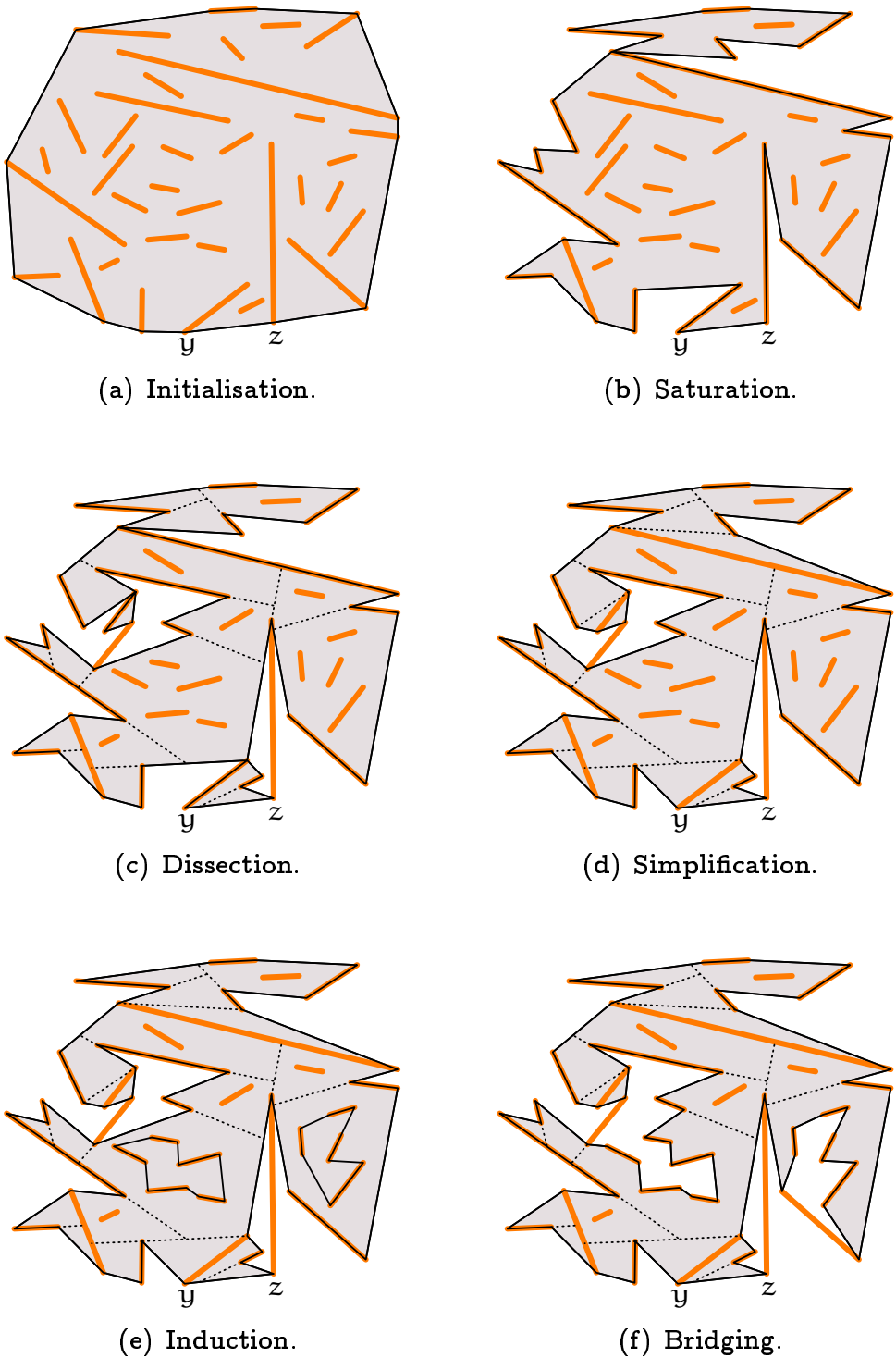
The rest of this chapter is organized as follows.

In Section 3.2 we list some properties of our working polygon P mentioned above by introducing a class of polygons called *frames*. Section 3.3, Section 3.4, and Section 3.5 describe the Saturation, Dissection, and Simplification Phase, respectively. Section 3.6 revisits and refines the Saturation algorithm according to the needs of the final Bridging Step. Then Section 3.7 provides a summary of the discussion in the four preceding sections and describes the final algorithm for Saturation, Dissection, and Simplification. Finally, in Section 3.8 we apply this algorithm to prove the existence of a Hamiltonian polygon inductively. We conclude by providing a short runtime analysis in Section 3.9, and give some final remarks regarding Hamiltonian polygons for other types of input objects in Section 3.10.

3.2 Frame Polygons

The central object in the algorithm is a polygon which initially is just the convex hull of the input segments but in the end becomes the Hamiltonian polygon that is the final goal of the whole construction. In this section we formally define the properties of these polygons that we call *frames*. We will make sure that the intermediate polygons belong to this class throughout the algorithm.

Definition 3.3 *A simply-connected polygon* $P = (V, E)$ *is called* **frame** *for a set* S *of disjoint line segments if and only if it has the following properties.*



(a) Initialisation. (b) Saturation.

(c) Dissection. (d) Simplification.

(e) Induction. (f) Bridging.

Figure 17: *Phases in the construction of a Hamiltonian polygon.*

(F1) $V(S) \subset \mathcal{R}(P)$;

(F2) P *is a subgraph of* $\mathrm{Vis}(S)$;

(F3) no vertex $v \in V$ *appears more than twice in* $P_{\circlearrowleft}$;

(F4) if a vertex $v \in V$ *appears twice in* $P_{\circlearrowleft}$ *then the angular domain around* v *intersects* P° *in two strictly convex angles. That is, for*

$$P_{\circlearrowleft} = (\ldots, p_i = v, \ldots, p_j = v, \ldots)$$

both $\measuredangle(p_{j\oplus 1}, v, p_{i\ominus 1})$ *and* $\measuredangle(p_{i\oplus 1}, v, p_{j\ominus 1})$ *are strictly convex;*

(F5) if for some $\overline{uv} \in S$ *it is* $u \in V$ *and* $v \notin V$ *(hence* $v \in P^\circ$ *by (F1) and (F2)) then* u *appears only once in* $P_{\circlearrowleft}$ *and* $\measuredangle_P u$ *is convex.*

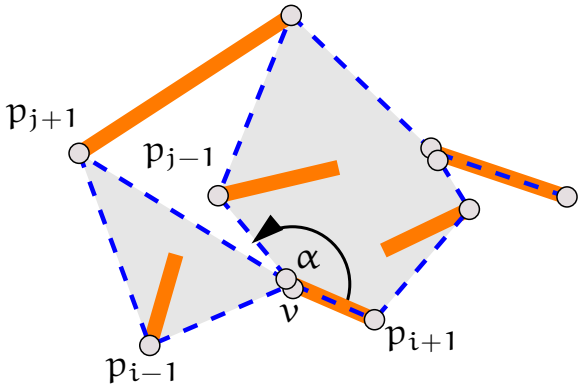
We call a vertex $v \in V$ *that appears twice in* $P_{\circlearrowleft}$ *a* **double vertex** *of* P*, while a vertex that appears only once in* $P_{\circlearrowleft}$ *is referred to as a* **single vertex** *of* P.

Figure 18 illustrates the different properties listed in Definition 3.3. In the figures, double vertices are usually indicated by two slightly shifted circles to increase their visibility; formally both occurrences of a double vertex correspond to the same point in the plane.

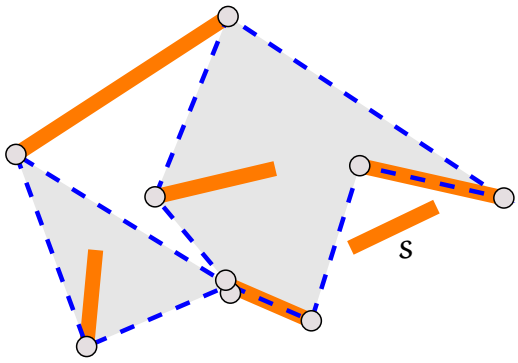
Observe that $\mathrm{conv}(\partial S)$ is always a frame for S. Properties (*F4*) and (*F5*) may look a bit strange at first sight, but they allow us to control number and type of reflex vertices that appear in the frame. Intuitively speaking, convex angles are desirable given that we are heading for a convex dissection in the end.

As Property (*F3*) suggests, the tour $P_{\circlearrowleft}$ of a frame P may visit segment endpoints twice. But in the end we are interested in a simple polygon, that is, we have to make sure that for every double vertex one of the occurrences in $P_{\circlearrowleft}$ can be eliminated in Phase 4 of the algorithm. Hence let us look at these vertices more closely.

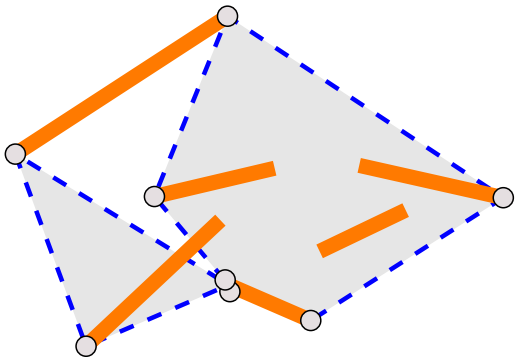
Proposition 3.4 *Consider a double vertex* b *of a frame* P. *Then either* b *appears at least once as a reflex vertex in* $P_{\circlearrowleft}$ *or* b *appears twice as a flat vertex in* $P_{\circlearrowleft}$.



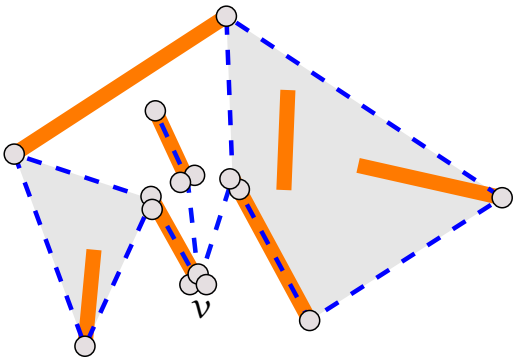
(a) A frame.



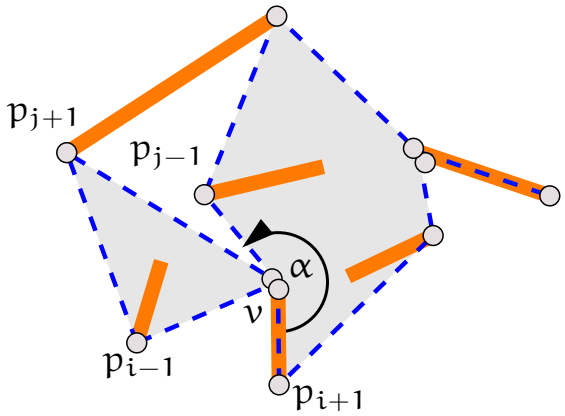
(b) Not a frame: violates (*F1*) because $V(s) \not\subseteq \mathcal{R}(P)$ for the segment s.



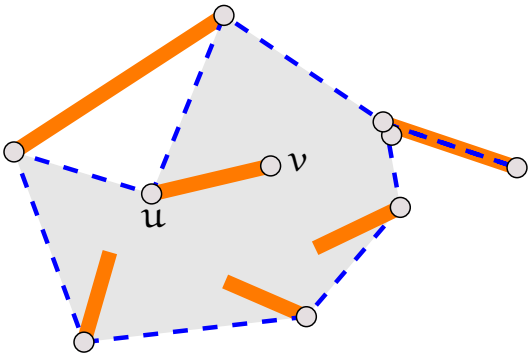
(c) Not a frame: violates (*F2*).



(d) Not a frame: violates (*F3*) as vertex v appears three times in $P_{\circlearrowleft}$.



(e) Not a frame: violates (*F4*) because $\alpha > \pi$.



(f) Not a frame: violates (*F5*) as vertex u is reflex and $v \in P^{\circ}$.

Figure 18: *Examples for (non-)frames.*

Proof. According to Property (*F3*) no vertex of P may appear more than twice in $P_{\circlearrowleft}$. Consider a double vertex b and let

$$P_{\circlearrowleft} = (\ldots, a, b, c \ldots, d, b, e, \ldots)$$

such that $\measuredangle(c, b, a)$ is not reflex.

If $\deg_P(b) = 2$ then b is incident to two cut-edges of P, that is, $c = d$ and $a = e$. Clearly either $\measuredangle(c, b, a)$ or $\measuredangle(e, b, d) = \measuredangle(a, b, c)$ is reflex or both angles are flat.

If $\deg_P(b) = 3$ then b is incident to exactly one cut-edge in P and without loss of generality $c = d$ and $a \neq e$. If $\measuredangle(c, b, e) > \measuredangle(c, b, a)$ then $\overline{cb}$ is a cut-edge of P that is not incident to the infinite face, in contradiction to the fact that P is a polygon. Hence $\measuredangle(c, b, e) < \measuredangle(c, b, a) \leq \pi$ and $\measuredangle(e, b, d) = \measuredangle(e, b, c)$ is reflex.

Otherwise we have $\deg_P(b) = 4$ and by Proposition 2.5 it is either $d, e \in \mathcal{D}_{(\measuredangle)}(c, b, a)$ or $d, e \in \mathcal{D}_{(\measuredangle)}(a, b, c)$.

Consider the case $d, e \in \mathcal{D}_{(\measuredangle)}(a, b, c)$. Then by Proposition 2.11 the circular order of a, c, d, e around b is a, e, d, c as shown in Figure 19(a). In particular, both $\mathcal{D}_{(\measuredangle)}(d, b, c) \cap \mathcal{B}_\varepsilon(b) \subset \mathrm{ext}(P)$ and $\mathcal{D}_{(\measuredangle)}(a, b, e) \cap \mathcal{B}_\varepsilon(b) \subset \mathrm{ext}(P)$, for sufficiently small $\varepsilon > 0$. Thus the curve that traverses the closed non-crossing subtrail $(b, c, \ldots, d, b)$ of $P_{\circlearrowleft}$ uniformly is not nullhomotopic, in contradiction to the fact that P as a frame is simply-connected.

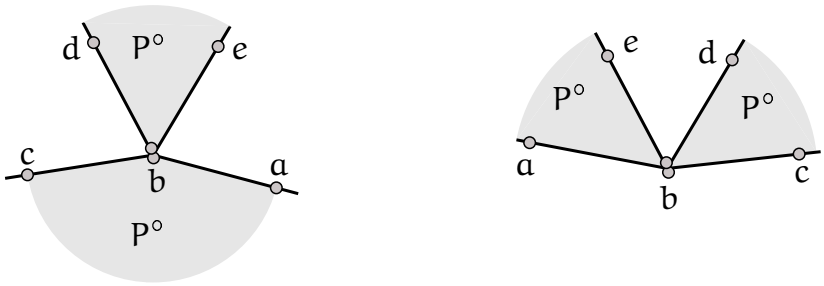
On the other hand, if $d, e \in \mathcal{D}_{(\measuredangle)}(c, b, a)$ then by Proposition 2.11 the circular order of a, c, d, e around b is a, c, d, e as shown in Figure 19(b). In particular b appears as a reflex vertex within (d, b, e).

In summary we have shown that b appears at least once as a reflex vertex in $P_{\circlearrowleft}$, unless b is incident to two collinear cut-edges of P. □

Note that a vertex may appear twice as a reflex vertex in a frame as the example in Figure 18(a) shows. But Property (*F4*) puts certain restrictions on the geometric situation around such a vertex.

3.3 Saturation

This section focuses on Phase 2 of the algorithm. The goal of this phase is to modify the current frame P locally in order to include the other



(a) $d, e \in \mathcal{D}_{(\measuredangle)}(a, b, c)$. (b) $d, e \in \mathcal{D}_{(\measuredangle)}(c, b, a)$.

Figure 19: *A double vertex appears once as a reflex vertex.*

endpoint of those segments for which only one endpoint appears as a vertex of P. The segments that lie completely in the interior of P are not handled in the Saturation Phase, although some of them might be "caught" by P accidentally. In general the frame will become a non-simple polygon during this phase, that is, some double vertices may be created.

Consider a segment $\overline{p_i q} \in S$ for which $p_i \in V(P)$ and $q \notin V(P)$. In the following we call such a segment *unsaturated* by P. Analogously a segment that is not unsaturated by P is referred to as *saturated*. Similarly a vertex in $V(P)$ is called *(un-)saturated* if its incident segment from S is (un-)saturated by P. Due to Property (*F5*) we know that for every unsaturated segment $\overline{p_i q} \in S$ its endpoint p_i is a single vertex of P such that $\measuredangle_P p_i$ is convex.

The plan is to integrate q as a vertex into P by using the edge $\overline{p_i q}$ and from q going back to P via a geodesic. In fact, there are two possible ways to continue from q: towards $p_{i\ominus 1}$ or towards $p_{i\oplus 1}$. Therefore we assign an orientation to each vertex in $P_{\circlearrowleft}$.

Definition 3.5 *For a frame polygon P, an* **orientation** *$u(P)$ is a function $u : P_{\circlearrowleft} \to \{-1, +1\}$. For a vertex p_i of $P_{\circlearrowleft}$ denote by*

$$v_u(p_i) := \begin{cases} p_{i\ominus 1} & , u(p_i) = -1, \\ p_{i\oplus 1} & , u(p_i) = +1. \end{cases}$$

the vertex **towards** *which p is* **oriented**.

The orientation of a vertex determines which incident edge of P is replaced when we saturate a possibly incident unsaturated segment using

the following local operation. In the figures, the orientation of a vertex will often be indicated by a small plus– or minus-symbol inside the disc that represents the vertex.

Operation 1 (Build-Cap(P, u, p_i)) *(Figure 20)*

Input: a frame P, an orientation $u(P)$, and an unsaturated segment $\overline{p_i q} \in S$ for which $p_i \in V(P)$.

Operation: Obtain P' from P by replacing the edge (p_i, r) by the path $(p_i, q) \cdot \mathrm{geo}(q, p_i, r)$, where $r := v_u(p_i)$. Set $u(p) := u(p_i)$ for all interior vertices of $\mathrm{geo}(q, p_i, r)$ including q.

Output: (P', u).

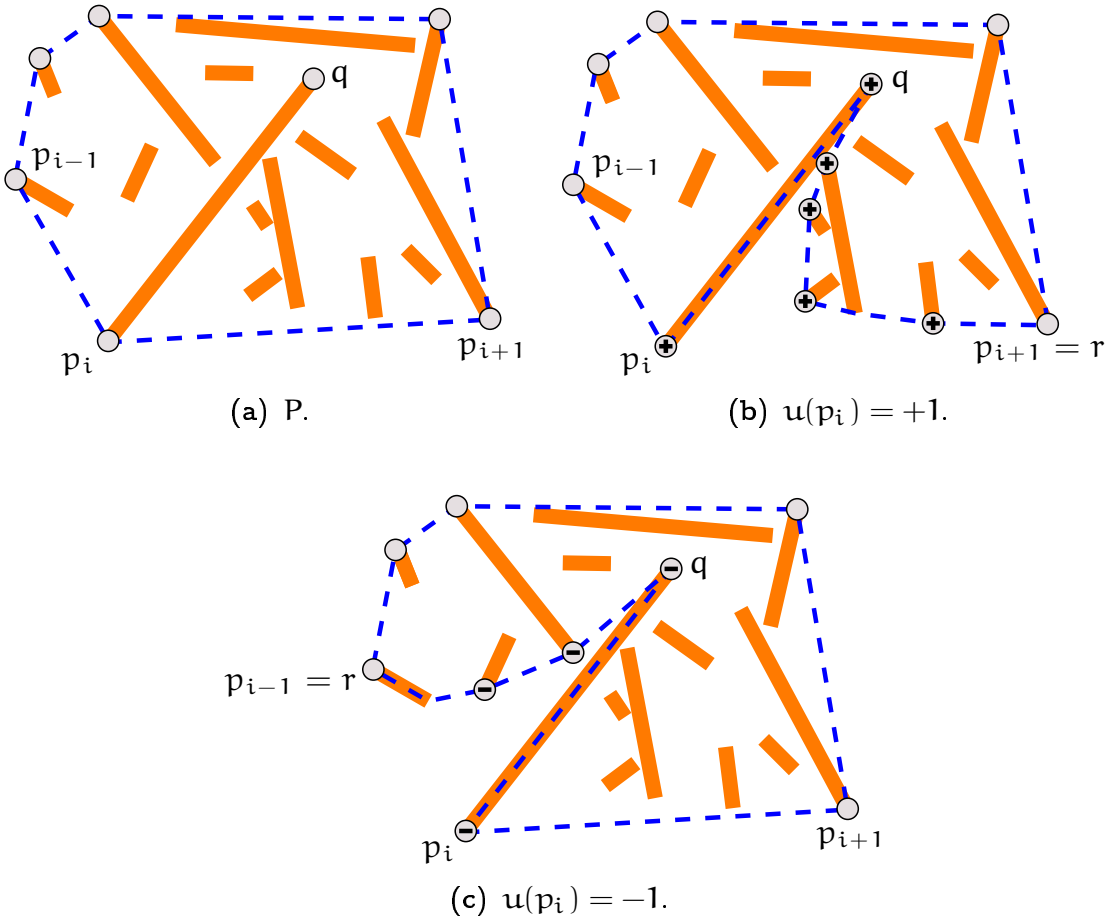


Figure 20: *A frame P and the result of Build-Cap(P, u, p_i) for the two possible orientations of p_i.*

Later we will use the additional degree of freedom provided by the orientation to make sure that a certain fixed edge of the initial convex

hull frame will never be replaced in the course of our construction. As explained in Section 3.1 this is important for the induction step. For now we just work with a uniform orientation $u \equiv +1$.

Note that P' is not necessarily simple because some of the interior vertices from $\mathrm{geo}(q, p_i, r)$ might already have been in $V(P)$.

Proposition 3.6 *The output P' of Build-Cap is a frame.*

Proof. Let us first convince ourselves that P' is a polygon to start with. By Property (*F5*) p_i is a convex single vertex of P and thus both angles $\measuredangle(q, p_i, p_{i\ominus 1})$ and $\measuredangle(p_{i\oplus 1}, p_i, q)$ are strictly convex. Hence p_i cannot occur as a vertex of $\mathrm{geo}(q, p_i, r)$, in particular p_i and q are single vertices of P' and $\{p_i, q\}$ is not a cut-edge of P'. As all edges inserted into P' along $\mathrm{geo}(q, p_i, r)$ bound the infinite face, P' is a polygon. Clearly P' is simply-connected.

(*F1*) and (*F2*) follow directly from the definition of geodesics and from the fact that the input polygon P is a frame. It remains to check properties (*F3*)–(*F5*).

Let $\mathrm{geo}(q, p_i, r) = (q = q_0, \ldots, q_k = r)$, for some $k \in \mathbb{N}$. We will argue separately for each of the vertices whose neighborhood changes from P to P'.

p_i: As argued above p_i is a single vertex in P'. Thus (*F3*) and (*F4*) hold. As the segment from S incident to p_i is an edge of P', Property (*F5*) is fulfilled as well.

q: argue as for p_i.

r: Clearly r appears as often in ${P'}_{\circlearrowleft}$ as in $P_{\circlearrowleft}$ and $\measuredangle_{P'} r < \measuredangle_P r$. This implies (*F3*)–(*F5*).

q_j, for $1 \leq j < k$: Vertex q_j is inserted as a convex vertex, that is, $\measuredangle_{P'} q_j$ is convex and its reflex domain $\mathcal{D}_{(\measuredangle)}(q_{j\ominus 1}, q_j, q_{j\oplus 1})$ is interior to P and exterior to P' in some sufficiently small neighborhood of q_j. The "exterior to P'"-part readily implies (*F4*), while the "interior to P"-part tells us together with Property (*F4*) of P that if q_j appears in $P_{\circlearrowleft}$ then it is a single reflex vertex. Properties (*F3*) and (*F5*) follow immediately.

□

The Saturation Phase is summarized in the operation below. It consists of iterated applications of Build-Cap to the current frame P until all segments are saturated by P. At this point every segment from S either has both endpoints in $V(P)$ or it lies completely in P°.

Operation 2 (Saturate(P, u))

Input: a frame P and an orientation $u(P)$.

Operation: As long as there exists an unsaturated vertex $p_i \in V(P)$, let $(P, u) \leftarrow$ Build-Cap(P, u, p_i).

Output: (P, u).

The following is an immediate consequence of Proposition 3.6.

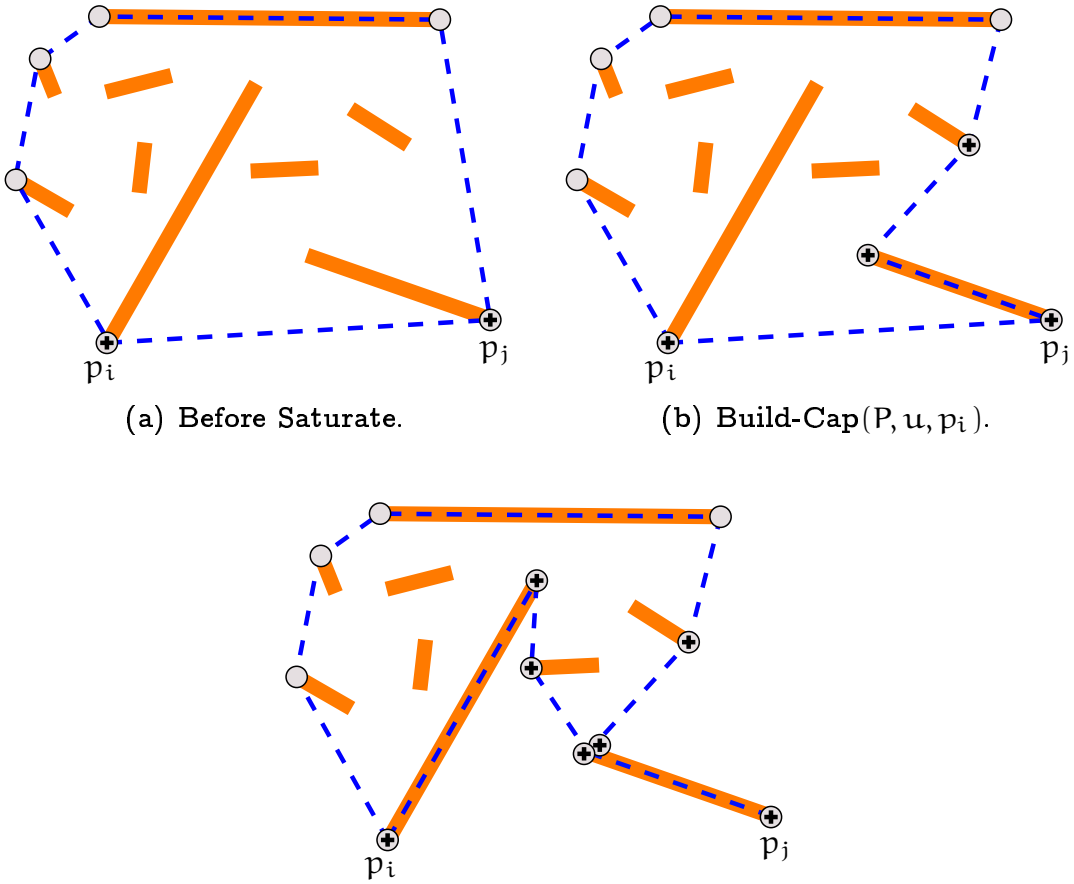
Corollary 3.7 *The output of Saturate is a frame.* □

The example depicted in Figure 21 shows that Saturate may create double vertices as well as cut-edges in the frame. This is not a problem, as these features will be dealt with in the Simplification Phase.

At this point we could end this section, as the description of the algorithm for the Saturation Phase is complete. However, in anticipation of the coming phases let us analyze the number and type of reflex vertices that Saturate may create. Reflex vertices are important as they may be revisited by a geodesic resulting in a double vertex of the frame. As we have to deal with double vertices during the Simplification Phase, we need to have some control on them.

Definition 3.8 *For a polygon P a subpath $(p_i, \ldots, p_k)$ of $P_{\circlearrowleft}$ is a* **reflex twin** *if and only if p_i and p_k are both reflex in $P_{\circlearrowleft}$ and p_j is flat, for all $i < j < k$.*

Clearly the initial frame $\mathrm{conv}(\partial S)$ does not contain any reflex twin. (It does not have *any* reflex vertices.) Build-Cap produces exactly one new reflex vertex: at q. Observe that the vertices on $\mathrm{geo}(q, p_i, r)$ are oriented "away" from this new reflex vertex. Together with a suitable orientation this avoids reflex twins. More precisely we can bound the number of reflex twins in terms of the number of alternations in the orientation.



(a) Before Saturate.

(b) Build-Cap(P, u, p_i).

(c) Build-Cap(P, u, p_j).

Figure 21: *Saturate may create cut-edges in the frame.*

Definition 3.9 *An* **alternation** *in an orientation* u *of a frame* P *is a pair of two consecutive vertices* $(p_i, p_{i\oplus 1})$ *in* $P_{\circlearrowleft}$ *such that* $u(p_i) = +1$, $u(p_{i\oplus 1}) = -1$, *and both* p_i *and* $p_{i\oplus 1}$ *are unsaturated.*

Proposition 3.10 *Consider a frame* P *with an orientation* u. *If Saturate is applied to* (P, u) *then in the resulting frame* P' *there is at most one new reflex twin for each vertex* v *in* $P_{\circlearrowleft}$ *such that*

1) v *is unsaturated and oriented towards a reflex vertex* r *in* $P_{\circlearrowleft}$; *in this case* r *is part of the reflex twin (if it is created);*

2) or v *is part of an alternation in* u *and* $u(v) = +1$.

(New twins in P' *are those which are not also present in* P.*)*

Proof. Consider a single Build-Cap and denote the participating vertices as above by p_i, q, and r. Let Q denote the input frame and let Q' denote the output frame of this particular Build-Cap operation. Assume without loss of generality $u(p_i) = +1$.

The operation creates exactly one new reflex vertex, namely at q. Vertex q has two neighbors in Q' one of which is p_i. By Property (*F5*) p_i is a convex vertex in $Q_{\circlearrowleft}$ and because $\measuredangle_{P'} p_i < \measuredangle_P p_i$ it is strictly convex in ${Q'}_{\circlearrowleft}$. If $\mathrm{geo}(q, p_i, r) \not\subseteq \overline{qr}$ then there is an interior vertex of the geodesic that is strictly convex in ${Q'}_{\circlearrowleft}$. (It might be a double vertex, but the particular occurrence along the geodesic is always convex.)

It remains to consider the case that $\mathrm{geo}(q, p_i, r) \subseteq \overline{qr}$. Suppose that r is reflex in $Q_{\circlearrowleft}$, that is, p_i is oriented towards a reflex vertex.

If r was already present in $P_{\circlearrowleft}$ then there must have been an unsaturated vertex oriented towards it: Consider the point where p_i became a neighbor of r. This might already be the case in $P_{\circlearrowleft}$. If not then p_i is visited by a geodesic in some Build-Cap operation applied to a vertex p_k. For p_i to become a neighbor of r as a result of this operation, p_k must be oriented towards r. Hence we may conclude inductively that there is a vertex v in $P_{\circlearrowleft}$ that is oriented towards r and the potential reflex twin (p_i, r) is covered by Condition 1.

Otherwise r has been created in this Saturate step by a previous Build-Cap operation. At that point r was endpoint of an unsaturated segment $\overline{p_j r}$ for which p_j was a vertex of the frame and $u(p_j) = -1$. Consider the other ($\neq p_j$) neighbor p_k of r directly after this Build-Cap

operation. If p_k is saturated at this point or if $u(p_k) = -1$ then the edge $\overline{p_k r}$ remains an edge of the frame throughout this Saturate step, in contradiction to p_i being an unsaturated neighbor of r at some point. Therefore it is $u(p_k) = +1$ and we found an alternation in the frame. As Build-Cap preserves orientations, this alternation must have been present in $P_{\circlearrowleft}$ already and it is destroyed at the point where r is visited and hence saturated. Instead from this point on there is a vertex that is oriented towards a reflex vertex in the frame which may lead to at most one new reflex twin as explained above. □

In particular, Proposition 3.10 together with Corollary 3.7 implies that the result of Saturate applied to the frame $\mathrm{conv}(\partial S)$ with a uniform orientation $u \equiv +1$ is a frame that does not contain any reflex twin.

3.4 Dissection

This section discusses how to dissect the frame P into convex polygons that eventually can be processed inductively. The dissection of the current frame P will be denoted by $\mathcal{D}$. As for the polygon P there are also several properties that we demand from $\mathcal{D}$.

Definition 3.11 *Consider a frame P for a set S of disjoint segments. A dissection $\mathcal{D}$ of P is* **nice** *if and only if*

- *(D1) every polygon $D \in \mathcal{D}$ is either simple or a line segment;*
- *(D2) every polygon $D \in \mathcal{D}$ has a common edge with P;*
- *(D3) in no polygon $D \in \mathcal{D}$ there is a reflex twin;*
- *(D4) for every reflex vertex r of some $D \in \mathcal{D}$ there is an incident edge $\overline{rq}$ that is common to both D and P and such that q is a convex vertex of D;*
- *(D5) for every $s \in S$: If $s \subset P^\circ$ then there is a $D \in \mathcal{D}$ such that $s \subset D^\circ$. Otherwise $s \cap D^\circ = \emptyset$, for all $D \in \mathcal{D}$.*

Recall that we begin with a trivial dissection $\mathcal{D}$ of the initial frame $\mathrm{conv}(\partial S)$, that is, $\mathcal{D} = \{P\}$. Clearly this initial dissection fulfills (*D1*)–(*D4*). Our goal is to achieve and maintain a nice dissection of P during

the Dissection Phase. For this we may have to update $\mathcal{D}$ already during the Saturation Phase. If edges of the frame P are modified that are also edges of a dissection polygon D then we just apply the same modifications to D as we do to P.

3.4.1 Canonical Dissections

As the frame might become non-simple during the Saturation Phase, we have to do something in order to ensure (*D1*). Consider a double vertex q of P. According to Property (*F4*) the angular domain around q intersects P° in two strictly convex angles. As we are heading for a convex dissection this seems a good point to split the dissection polygon that contains q into two polygons at q. Doing so for each double vertex created obviously yields a dissection into simple polygons and line segments, where the line segments correspond to cut-edges of P. Clearly Property (*D2*) is satisfied as well while (*D3*) and (*D4*) are an easy consequence of Proposition 3.10.

But in order to ensure (*D5*) we have to consider another type of events that might occur during the Saturation Phase or even already from the very beginning: the frame can have segments from S as diagonals. Let us refer to such segments as *segment diagonals*. Segment diagonals may appear in the beginning as diagonals of the convex hull (Figure 22(a)), and during Build-Cap if an endpoint of an unsaturated segment is visited by a geodesic (Figure 22(b)) or a geodesic visits both endpoints of a segment that was in the interior of the frame before (Figure 22(c)).

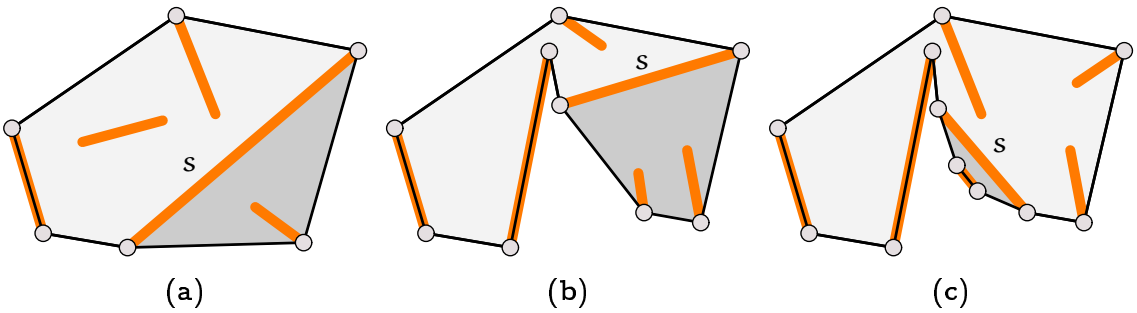


Figure 22: *A segment diagonal* s *and the resulting canonical dissections.*

As a segment diagonal is not contained in the interior of the frame

it should not intersect the interior of any dissection polygon according to $(D5)$. This can be achieved easily if we split the dissection polygons along all segment diagonals. This additional separation makes also sense if we keep in mind our final goal of convex dissection polygons: By Property $(F5)$ and the way geodesics are used in Build-Cap the endpoints of a segment diagonal are convex vertices of both dissection polygons they appear in. Let us summarize these observations in the following proposition.

Proposition 3.12 *We can achieve a nice dissection of the frame at the end of the Saturation Phase by splitting the dissection polygons at all double vertices and along all segment diagonals.*

Proof. Properties $(D1)$–$(D4)$ are clear as discussed above. For $(D5)$ note that at the end of the Saturation Phase there is no unsaturated segment. That is, each segment from S either is in the interior of the frame P or both of its endpoints are in $V(P)$. In the latter case the segment is either an edge of P or a segment diagonal, and in both cases it is disjoint from the interior of each dissection polygon. In the former case, recall that the segments from S are disjoint. In particular, no segment can contain a double vertex in its relative interior nor can it cross a segment diagonal. This proves $(D5)$. □

Definition 3.13 *Consider a frame P and a dissection $\mathcal{D}$ of P. The* **canonical dissection** *of $\mathcal{D}$ with respect to P is obtained from $\mathcal{D}$ as follows: Split the dissection polygons at all double vertices of P and along all segment diagonals.*

Note that the segment endpoints appear both as vertices of the frame P and as vertices of the dissection polygons. Hence, when we talk about a "convex vertex" we have to specify whether we refer to it as a convex vertex of the frame or of one of the dissection polygons. The following proposition characterizes the reflex vertices of the canonical dissection polygons.

Proposition 3.14 *Consider a frame P for S and a polygon D from a dissection $\mathcal{D}$ of P that fulfills $(D1)$ and $(D4)$. Then every reflex vertex of D is single in $P_{\circlearrowleft}$.*

Proof. Consider a reflex vertex p_i of some $D \in \mathcal{D}$. By Property (*D4*) p_i is also a vertex of $P_{\circlearrowleft}$. On the other hand, by Property (*F4*) and (*D1*) no double vertex of P can be reflex in any $D \in \mathcal{D}$. □

3.4.2 Extension to Interior Segments

Clearly the polygons in the canonical dissection are not necessarily convex. A first idea to obtain a dissection into convex polygons from $\mathcal{D}$ is the following: choose a reflex vertex p_i of some $D \in \mathcal{D}$ and draw a ray from p_i that splits $\measuredangle_D p_i$ into two strictly convex angles. Go along the ray until it hits ∂D or a previously drawn ray at some point x. If the segment $\overline{p_i x}$ does not cross any segment from S then we refine the dissection by splitting D along $\overline{p_i x}$. Note that x is not necessarily endpoint of a segment from S. The resulting dissection depends on the order in which the rays are drawn, but any order would do at this point.

But if any of the segments $\overline{p_i x}$ crosses a segment s from S, splitting D along $\overline{p_i x}$ would violate (*D5*). Hence we handle this case in a different way, by extending P to incorporate s. This leads to a new basic operation, Extend-Reflex.

Operation 3 (Extend-Reflex$(P, u, \mathcal{D}, p_i, r, \overrightarrow{s})$**)** *(Figure 23)*

Input: a frame P, an orientation $u(P)$, a dissection $\mathcal{D}$ of P, a reflex vertex p_i of some $D \in \mathcal{D}$, a convex vertex r of D, and a ray $\overrightarrow{s}$ emanating from p_i.

Preconditions: $\overline{p_i r}$ is a common edge of D and P, $\overrightarrow{s}$ cuts $\measuredangle_D p_i$ into two strictly convex angles, and $\overrightarrow{s}$ hits[1] the segment $\overline{qt}$ from S that lies in D° at a point $x \in \overline{qt}^{\ominus}$.

Operation: (Assume without loss of generality that r and t are on the same side of the supporting line of $\overrightarrow{s}$.) Obtain P' from P by replacing the edge $\overline{p_i r}$ by the path $\mathrm{geo}(p_i, x, q) \cdot (q, t) \cdot \mathrm{geo}(t, x, p_i, r)$. Dissect $\mathcal{D}$ canonically. Set $u(\cdot) := -1$ for all interior vertices of $\mathrm{geo}(p_i, x, q)$ including q, and $u(\cdot) := +1$ for all interior vertices of $\mathrm{geo}(t, x, p_i, r)$ including t.

Output: $(P', u, \mathcal{D})$.

[1] More precisely, $\overline{p_i x}^{\ominus} \cap (\partial S \cup \partial D) = \emptyset$.

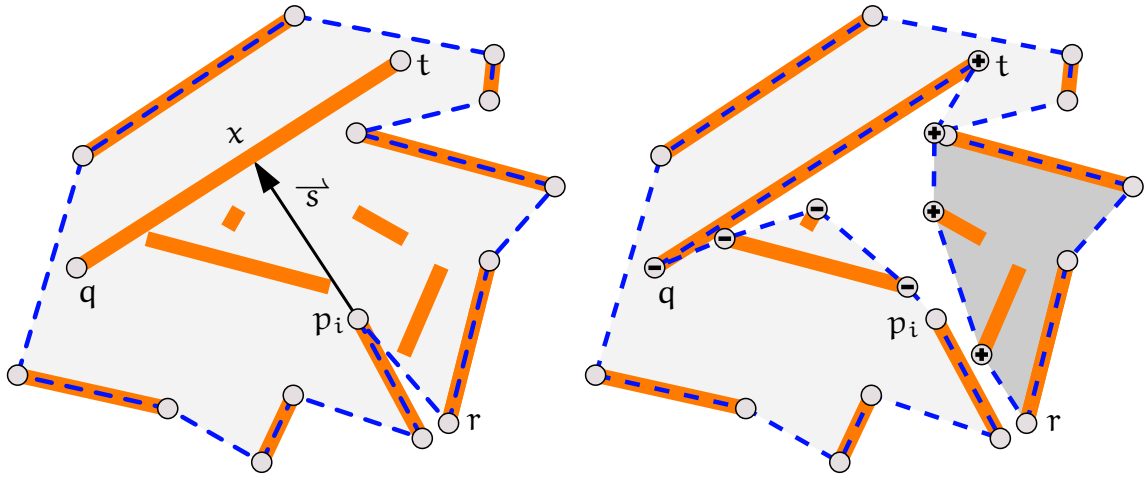


Figure 23: *Extend-Reflex*$(P, u, \mathcal{D}, p_i, r, \overrightarrow{s})$.

There are two variants of Extend-Reflex, depending on whether r follows or precedes p_i in $P_{\circlearrowleft}$. We have described only the first above and refer to this variant in the following. The other variant is completely symmetric. Looking at the preconditions of Extend-Reflex we have to ensure that for each reflex vertex of a polygon D from $\mathcal{D}$ there is an incident edge that is common to both D and P and whose other endpoint is a convex vertex of D. This is exactly what Property $(D4)$ demands.

Strictly speaking the geodesics above are not well defined because the point x is not a vertex of $\mathrm{Vis}(S)$. But since $\{q, t\}$ is an edge of $\mathrm{Vis}(S)$ and the piecewise linear curves (p_i, x, q) and (t, x, p_i, r) do not penetrate any segment from S the definition carries over.

As discussed in Section 2.6, the geodesic of a size four path is not necessarily simple. Also in Extend-Reflex one endpoint of the segment hit by the ray $\overrightarrow{s}$ can go up from degree zero to degree four in P (Figure 24). Nevertheless, we will show below that the resulting polygon is always a frame. As Figure 23 demonstrates and as we have already seen for Build-Cap, Extend-Reflex might create some new unsaturated segments which prevent the dissection from being nice. This problem is easily resolved by applying once again Saturate to the frame that results from Extend-Reflex. The orientation along the geodesics is chosen in such a way that no vertex of P' that is incident to an unsaturated segment is oriented towards a reflex vertex. Recall that as a precondition of Extend-Reflex r is a convex vertex of D. Hence by Proposition 3.10 there is at most one new reflex twin in all dissection polygons for the resulting frame: (q, t). (Clearly both endpoints of the segment we extend

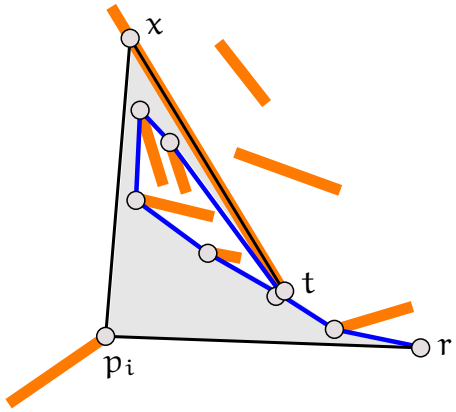


Figure 24: *Vertex* t *may have degree four after Extend-Reflex.*

to may be reflex in the result frame P' of Extend-Reflex.)

Another implication is that if we use Extend-Reflex then we will not be able to maintain a nice dissection of the frame throughout the Dissection Phase. However, the reflex twin (q, t) is the only one present in the whole frame. At this point we make use of the fact that we did not specify so far from which reflex vertex we shoot the ray. If we just choose one vertex of the reflex twin created by Extend-Reflex then this twin will no longer correspond to two consecutive reflex vertices along a polygon in $\mathcal{D}$ because the vertex from which the ray is shot will become convex in the process of handling the ray. The resulting dissection will not be nice in the strict sense, but it comes pretty close which gives rise to the following definition.

Definition 3.15 *Consider a frame* P *for a set* S *of disjoint segments. A dissection* $\mathcal{D}$ *of* P *is* **almost nice** *if and only if it fulfills (D1), (D2), (D4), and (D5) as well as the condition below.*

(D3-) The number of reflex twins counted over all polygons from $\mathcal{D}$ *is at most one.*

Note that there may be many reflex twins in $P_{\circlearrowleft}$, but for the moment we are concerned with those only that are also consecutive vertices within a dissection polygon. Let us discuss a few special situations that may arise in Extend-Reflex. First of all, if $\overline{p_i r}$ is a segment from S then it is an epigonal in P'. Second, several situations may lead to the creation of cut-edges within the frame.

Clearly a cut-edge may be created when vertices of a reflex twin appear consecutively along a geodesic. Actually, in the end we have to be

very careful with such events because they are potentially troublesome in the Simplification Step later, but for now this is not our concern. However, there are also a few other ways to introduce cut-edges in the frame which will be discussed below.

If p_i has a neighbor w in D that is a reflex vertex of D then w may appear as a second vertex on $\mathrm{geo}(p_i, x, q)$. In this case $\overline{p_i w}$ is a cut-edge of P' and $\deg_{P'}(p_i) = 1$. Similarly, if r has a neighbor v in D that is a reflex vertex of D then v might appear as a second vertex on $\mathrm{geo}(r, p_i, x, q)$ thereby reverting $\overline{rv}$ to a cut-edge of P'. An example illustrating both cases is shown in Figure 25.

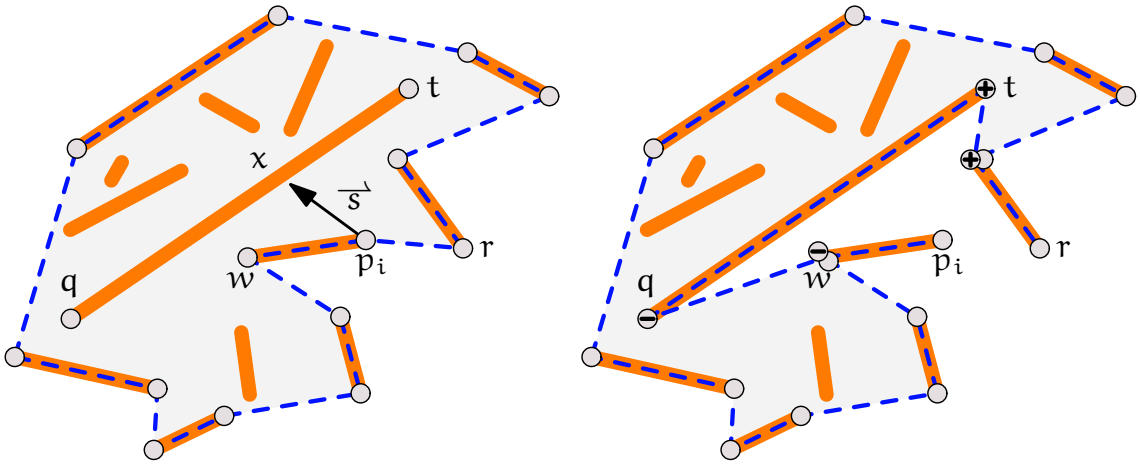


Figure 25: *Extend-Reflex may create cut-edges at* p_i *or* r.

There is another possibility to introduce a new cut-edge in P': if t appears twice as a vertex on $\mathrm{geo}(r, p_i, x, t)$ and the segments from S in the interior of the quadrilateral $\square(r, p_i, x, t)$ are collinear. In this case all these segments as well as the visibility edges connecting them along their underlying line are cut-edges in P', as shown in Figure 26.

The discussion of these degenerate cases is just for illustration; from an algorithmic point of view vertices incident to cut-edges are in no way more problematic than any other double vertices. Now we are ready to prove the promised invariant of Extend-Reflex.

Proposition 3.16 *If the input dissection $\mathcal{D}$ is almost nice then the output P' of Extend-Reflex is a frame.*

Proof. As for Build-Cap we will first argue that P' is a polygon. First note that the ray $\overrightarrow{s}$ is not collinear to $\overline{qt}$ as a precondition of Extend-

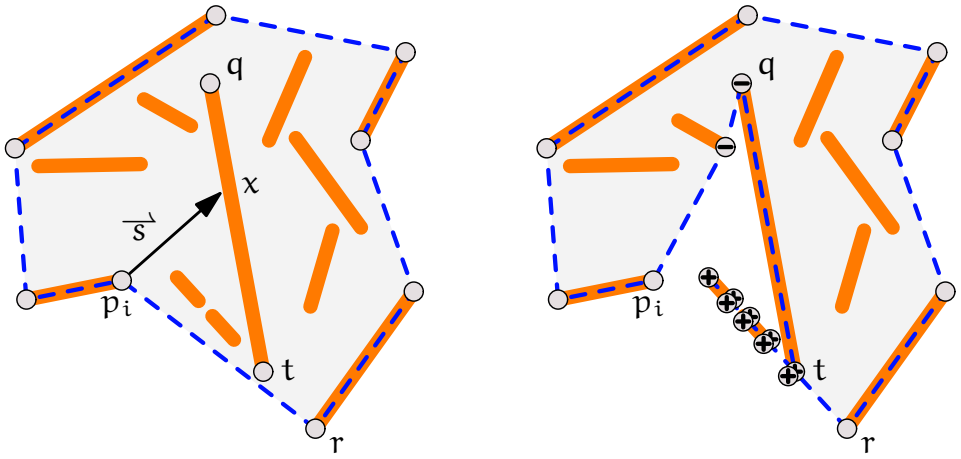


Figure 26: *Extend-Reflex may create many cut-edges along collinear segments on* $\mathrm{geo}(r, p_i, x, t)$.

Reflex. Hence the two geodesics in Extend-Reflex do not share a vertex and all edges of both geodesics as well as $\overline{qt}$ are incident to the infinite face of P'. Thus P' is indeed a polygon.

Clearly P' is simply-connected and (*F1*) as well as (*F2*) follow directly from the definition of geodesics and from the fact that the input polygon P is a frame. It remains to check properties (*F3*)–(*F5*).

For internal vertices of $\mathrm{geo}(p_i, x, q)$ and $\mathrm{geo}(t, x, p_i, r)$ one can argue as in Proposition 3.6. Hence, we have to consider the vertices p_i, r, q, and t only.

p_i: By Proposition 3.14 it is a single vertex of P and hence of P'. This implies (*F3*) as well as (*F4*). As p_i is a reflex vertex in $P_{\circlearrowleft}$ its incident segment from S is saturated in P by (*F5*) and hence (*F5*) also holds in P'.

r: As a convex vertex of D it cannot appear as an interior vertex on $\mathrm{geo}(t, x, p_i, r)$ even if it is a reflex vertex of the quadrilateral $\square(t, x, p_i, r)$. Thus r appears as often in $P'_{\circlearrowleft}$ as it did in $P_{\circlearrowleft}$ and $\measuredangle_{P'} r < \measuredangle_P r$ which implies (*F3*)–(*F5*).

q: The incident segment $\overline{qt}$ is an edge of P' which implies (*F5*). As q is a single vertex in P', both (*F3*) and (*F4*) hold trivially.

t: The incident segment $\overline{qt}$ is an edge of P' which implies (*F5*). If t appears twice on $\mathrm{geo}(t, x, p_i, r)$ it is a double vertex of P' and (*F4*) follows as in Proposition 3.6 because t appears as an internal

vertex of a geodesic. Otherwise t is a single vertex of P′ and (*F4*) is trivial. In both cases (*F3*) is fulfilled.

□

3.4.3 Preserving Common Edges

So far we have argued how to maintain a frame in case that one of the rays shot from a reflex vertex of a dissection polygon hits a segment that lies in its interior. Now it is time to address our second concern: how to maintain an almost nice dissection.

Indeed this is not granted by now, even if none of the rays drawn hits any interior segment: the dissection that results from repetitive splits along rays need not fulfill Property (*D2*) as the example in Figure 27 shows. We have to take into account that, whenever a ray hits the boundary of the current region and thus the region is split along this ray, the edge hit might have been the last common edge of P and one of the newly created regions.

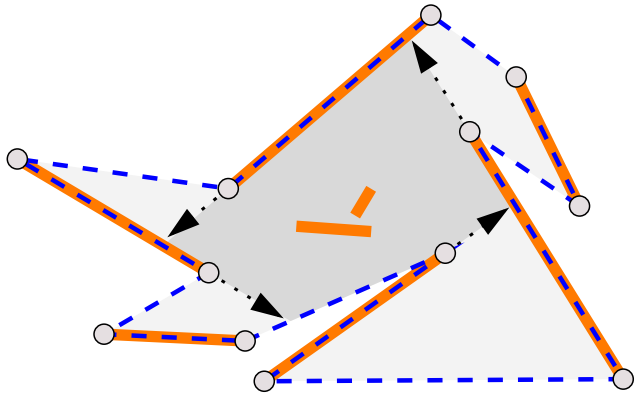


Figure 27: *The shaded dissection polygon violates (D2) as it does not have any common edge with the surrounding frame.*

In order to guarantee an almost nice dissection during the Dissection Phase we introduce another basic operation, Drag-Edge, that is applied under certain circumstances when the ray hits a common edge of the dissection polygon D and the frame P. The exact conditions when this operation is applied will be discussed below and we will show how to ensure a common edge between every D and P throughout the Dissection Phase.

Operation 4 (Drag-Edge$(P, u, \mathcal{D}, p_i, \overrightarrow{s}, \overline{p_k p_\ell})$**)** *(Figure 28)*

Input: a frame P with an orientation $u(P)$, a dissection $\mathcal{D}$ of P, a reflex vertex p_i of some $D \in \mathcal{D}$, a ray $\overrightarrow{s}$ emanating from p_i, and an edge $\overline{p_k p_\ell}$ of ∂D hit by $\overrightarrow{s}$ at a point $x \in \overline{p_k p_\ell}^{\ominus}$.

Preconditions: $\overline{p_k p_\ell}$ is a common edge of P and D, p_k is a convex vertex of D, and $\overrightarrow{s}$ cuts the angle $\measuredangle_D p_i$ into two strictly convex angles.

Operation: Obtain P' from P by replacing the edge $\overline{p_k p_\ell}$ by the path $\text{geo}(p_k, x, p_i) \cdot \text{geo}(p_i, x, p_\ell)$. Dissect $\mathcal{D}$ canonically. Set $u(\cdot) := -1$ for all interior vertices of $\text{geo}(p_k, x, p_i)$ and $u(\cdot) := +1$ for all interior vertices of $\text{geo}(p_i, x, p_\ell)$.

Output: $(P', u, \mathcal{D})$.

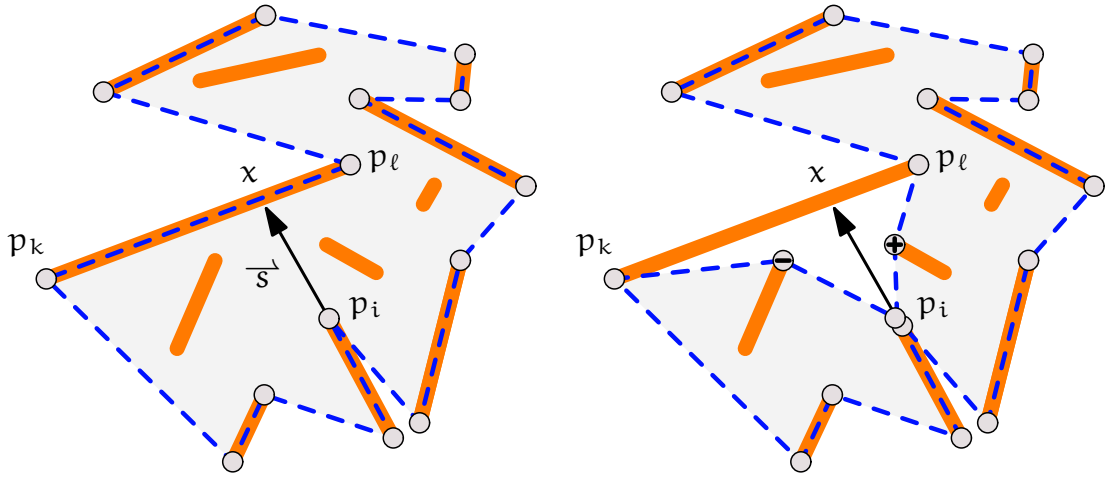


Figure 28: *Drag-Edge*$(P, u, \mathcal{D}, p_i, \overrightarrow{s}, \overline{p_k p_\ell})$.

Observe that Drag-Edge creates an epigonal of the frame in case that $\overline{p_k p_\ell}$ is a segment from S. Similar to Extend-Reflex there are a few situations in which Drag-Edge creates cut-edges within the frame.

If p_i has a neighbor w in D that is a reflex vertex of D then w might appear as a second vertex on $\text{geo}(p_i, x, q)$. This is the exactly the same situation as in Extend-Reflex, so we do not discuss it here again. In the final algorithm we will even avoid such a situation because it causes problems for the Simplification Phase later. But for the moment this is not an issue.

Apart from the usual possibility that a geodesic visits two vertices that form a reflex twin, there is another situation in which Drag-Edge creates a cut-edge in P': if the other ($\neq p_\ell$) neighbor of p_k in the dissection polygon D under consideration is a reflex vertex of D, it may appear as a second vertex on $\mathrm{geo}(p_k, x, p_i)$. One particular degenerate instance of this situation occurs if p_k is adjacent to p_i in D. In this case the edge $\overline{p_i p_k}$ which is already an edge of the frame may also appear on $\mathrm{geo}(p_i, x, p_k)$, as shown in Figure 29. Observe that this situation cannot occur for p_ℓ if it happens to be a reflex vertex of D.

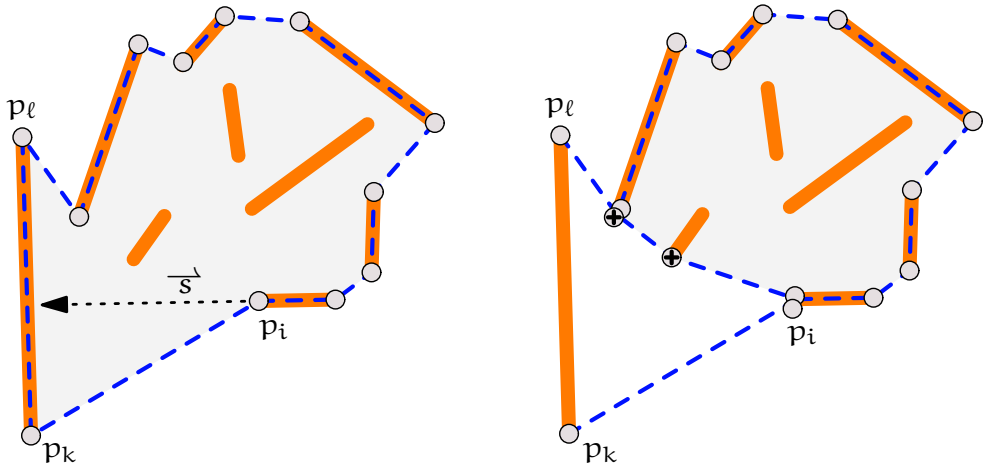


Figure 29: *Drag-Edge may create cut-edges at* p_k *and/or* p_ℓ.

As for the other basic operations we prove that Drag-Edge produces a frame.

Proposition 3.17 *If the input dissection $\mathcal{D}$ is almost nice then the output P' of Drag-Edge is a frame.*

Proof. Let us first argue that P' is a polygon. As the ray $\vec{s}$ hits $\overline{p_k p_\ell}$ in its relative interior, both are not collinear and the two geodesics share exactly one vertex: p_i. As $\overline{p_k p_\ell}$ is an edge of P all edges along both geodesics are incident to the infinite face of P'. Thus P' is indeed a polygon.

Clearly P' is simply-connected. It remains to check properties (*F1*)–(*F5*). (*F1*) and (*F2*) follow directly from the definition of geodesics and from the fact that the input polygon P is a frame.

For internal vertices of $\mathrm{geo}(p_k, x, p_i)$ and $\mathrm{geo}(p_i, x, p_\ell)$ one can argue as in Proposition 3.6. Hence, we have to consider the vertices p_i, p_k,

and p_ℓ only.

p_i: By Proposition 3.14 p_i is a reflex single vertex in P. Therefore its incident segment from S is saturated in P and hence in P' by (*F5*). As a single vertex of P vertex p_i appears twice in $P'_{\circlearrowleft}$ and thus (*F3*) holds as well. (*F4*) follows from the fact that $\overrightarrow{s}$ cuts the angle $\measuredangle_D p_i$ into two strictly convex angles.

p_k: Clearly p_k appears as often in $P'_{\circlearrowleft}$ as it did in $P_{\circlearrowleft}$ and $\measuredangle_{P'} p_k < \measuredangle_P p_k$ which implies (*F3*)–(*F5*).

p_ℓ: argue as for p_k.

□

3.4.4 The direction of $\overrightarrow{s}$

So far it may seem convenient to simply always apply Drag-Edge when a ray hits a common edge of P and the dissection polygon D under consideration. But as we will see in the discussion of the Simplification Phase, we cannot always apply Drag-Edge if we want to construct a simple polygon from the frame afterwards. The problematic part of Drag-Edge is that it creates a second reflex occurrence of the vertex p_i.

The reason to introduce Drag-Edge was to ensure Property (*D2*), that is, a common edge of each dissection polygon with the surrounding frame. In the light of this it is not really necessary to always apply Drag-Edge: suppose that at some point we shoot a ray $\overrightarrow{s}$ from a reflex vertex p_i of some dissection polygon D and both edges incident to p_i are common edges of P and D. If we then simply dissect D along $\overrightarrow{s}$, there is certainly still a common edge in both resulting dissection polygons.

On the other hand, we may have to be careful in the next iteration: Suppose that the ray $\overrightarrow{s}$ hits an edge incident to another reflex vertex p_k of D. If Drag-Edge is not applied then the ray $\overrightarrow{q}$ shot from p_k in the next iteration may hit back to $\overrightarrow{s}$ and thereby generate a dissection polygon that has no common edge with the surrounding frame P. See the example shown in Figure 30.

To prevent this situation, we make use of the fact that so far we did not specify precisely in which direction the ray from a reflex vertex p_i is shot, as long as it cuts the reflex angle at p_i into two strictly convex angles. We will see that it is sufficient to shoot the ray in direction of

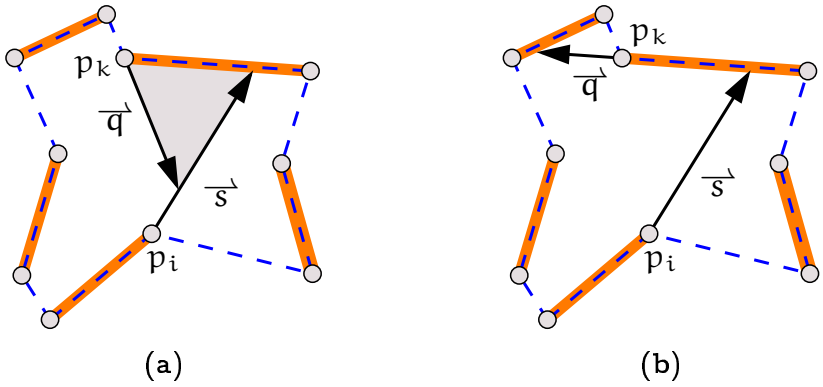


Figure 30: *The shaded dissection polygon in (a) has no common edge with the frame.*

the edge that was hit in the previous iteration, if any; see Figure 30(b) for an example.

3.4.5 A First Dissection Algorithm

Let us summarize the algorithm for the Dissection Phase as discussed above. Note that this is just a preliminary description. Later the algorithm will be adapted and refined to the needs of the Simplification Phase that is the subject of the next section.

Algorithm 3.18 *The following procedure maintains a frame* P *for* S *along with an orientation* u *and an almost nice dissection* $\mathcal{D}$ *of* P.

Start with $P \leftarrow \text{conv}(\partial S)$, $u \equiv +1$, and $\mathcal{D} \leftarrow \{P\}$ and repeat the following steps until at some point all polygons in $\mathcal{D}$ are convex after Step (a).

a) Apply Saturate(P, u) and dissect $\mathcal{D}$ canonically.

b) Choose (a preceding iteration might have made that choice already) a reflex vertex p_i of some $D \in \mathcal{D}$ and let $\overline{p_i r}$ be a common edge of P and D such that $\measuredangle_D r$ is convex. Cast a ray $\overrightarrow{s}$ from p_i that cuts $\measuredangle_D p_i$ into two strictly convex angles.

c) If $\overrightarrow{s}$ hits a segment $\overline{qt} \in S$ with $\overline{qt} \subset D^\circ$ then apply Extend-Reflex$(P, u, \mathcal{D}, p_i, r, \overrightarrow{s})$ and choose q as a next reflex vertex to shoot a ray from.

d) Else if $\overrightarrow{s}$ hits a common edge $\overline{p_k p_\ell}$ of P and D then

- If not both edges incident to p_i are common edges of P and D then apply Drag-Edge$(P, u, \mathcal{D}, p_i, \overrightarrow{s}, \overline{p_k p_\ell})$.
- If p_ℓ is reflex in D (assume without loss of generality that if p_k is reflex then also p_ℓ is reflex) then shoot the next fray from p_ℓ in direction of $\overrightarrow{p_k p_\ell}$.

e) Else split D along $\overrightarrow{s}$ and update $\mathcal{D}$ accordingly.

Although we have shown that each single operation preserves the frame properties, there are a few missing links because some of the corresponding propositions required the current dissection to be almost nice. For Saturate Corollary 3.7 asserts that the output is a frame. The next proposition states that Saturate also respects the current dissection $\mathcal{D}$ in a certain sense.

Proposition 3.19 *All edges constructed by Saturate in Step (a) of Algorithm 3.18 stay within the dissection polygon* D *that was modified in the preceding iteration. (For the first iteration let* $D := \text{conv}(\partial S)$*.)*

Proof. For the initial step where $\mathcal{D} = \{\text{conv}(\partial S)\}$ clearly all geodesics stay within the only dissection polygon $\text{conv}(\partial S)$. Consider a dissection polygon D that was modified in the previous iteration of Algorithm 3.18 and the Saturate step at begin of the next iteration.

Clearly none of the geodesics crosses an edge of D that is also (part of) an edge of the surrounding frame P. The only other option to leave D is by crossing an edge e of D that corresponds to a ray $\overrightarrow{a}$ that was shot from some reflex vertex in a previous step. Consider the first Build-Cap operation in the Saturate step where the constructed geodesic crosses e, and let $\overline{p_i q}$ be the segment saturated by this operation where p_i is already a vertex of the frame at that point and let $r = v_u(p_i)$.

First note that the ray that generated e was not shot from p_i because by Property (*F5*) all reflex vertices of a frame are saturated. As the interior of the polygon $\text{geo}(q, p_i, r) \cdot (p_i, q)$ is disjoint from ∂S and all interior vertices of $\text{geo}(q, p_i, r)$ are reflex in this polygon, the edge e crossing $\text{geo}(q, p_i, r)$ must intersect either $\overline{p_i q}$ or $\overline{p_i r}$. Clearly e does not intersect $\overline{p_i q}$: if a ray hits a segment in the interior of D then Extend-Reflex is applied. By assumption the construction of $\text{geo}(q, p_i, r)$ was the first time that a dissection edge was crossed by P; thus e does not cross $\overline{p_i r}$, either. Moreover, if e intersects $\overline{p_i r}$ then the ray $\overrightarrow{a}$ must have hit $\overline{p_i r}$. But this is impossible since the edge is incident to an

unsaturated segment and the ray shooting is applied only to frames in which all segments are saturated. Thus, $\mathrm{geo}(q, p_i, r)$ does not intersect e and all geodesics constructed in this Saturate step stay within D. □

It remains to analyze the properties of the dissection $\mathcal{D}$ in Algorithm 3.18.

Proposition 3.20 *P is always a frame and $\mathcal{D}$ is always an almost nice dissection of P after Step (a) in Algorithm 3.18.*

Proof. The statement is true for the first iteration of Algorithm 3.18 by Proposition 3.12 and Corollary 3.7. Consider an arbitrary iteration of Algorithm 3.18 and suppose the statement holds after Step (a). Let D denote the dissection polygon modified during this iteration.

If the frame is modified in this iteration then either Extend-Reflex or Drag-Edge is applied. As the current dissection is almost nice, both operations output a frame according to Proposition 3.16 and Proposition 3.17 which is preserved by the following Saturate Step by Corollary 3.7. If the frame is not modified then there are no new unsaturated segments and hence it is not modified in the following Saturate Step, either. Thus P is still a frame after Step (a) of the following iteration.

It remains to show that $\mathcal{D}$ stays an almost nice dissection. We have to check Properties (*D1*), (*D2*), (*D3-*), (*D4*), and (*D5*).

(*D1*) is an obvious consequence of the canonical dissection because the frame is non-simple at degree four vertices and cut-edges only. An edge generated by a ray properly cuts D into two simple polygons because the ray stops as soon as it hits ∂D.

For (*D3-*) observe that a reflex vertex of some $D \in \mathcal{D}$ is created by either Build-Cap or Extend-Reflex: If a ray hits an edge the resulting vertex is strictly convex on both sides, and Drag-Edge decreases the angles at both vertices of the edge that is hit by the ray. Both Extend-reflex and Drag-Edge orient the vertices along their constructed geodesics such that there is no alternation in the resulting orientation. (Recall that only endpoints of unsaturated segments count as an alternation.) Also, none of the vertices incident to an unsaturated segment is oriented towards a reflex vertex of any dissection polygon. Hence by Proposition 3.10 Saturate does not create any reflex twin in any dissection polygon. The only operation that may create such a reflex twin is Extend-Reflex. But then starting from the next iteration of Algo-

rithm 3.18, one of the vertices in the twin is not reflex in any dissection polygon anymore because the next ray is shot from this vertex. This proves (*D3-*).

Regarding (*D4*) note that all edges of dissection polygons are common with the frame before the first ray is shot. A reflex vertex a can only loose an incident common edge of its dissection polygon D with the frame if it is hit by a ray $\overrightarrow{s}$ shot from some other reflex vertex of D. But then either Drag-Edge is applied which once again creates an common edge of D and P incident to a or the next ray is shot from a and a is not a reflex vertex of any dissection polygon anymore at the end of the next iteration. In the meantime the other edge incident to a (the one that was not hit by $\overrightarrow{s}$) is still a common edge of P and D. By (*D3-*) not both vertices of the edge hit by $\overrightarrow{s}$ are reflex in D. Therefore (*D4*) continues to hold.

(*D2*) is certainly true before the first ray is shot. Whenever an operation (Extend-Reflex or Drag-Edge) is applied in an iteration, all resulting dissection polygons have common edges with the frame along the constructed geodesics. Hence suppose that no operation is applied in an iteration and the dissection polygon D is just dissected along a ray $\overrightarrow{s}$ shot from some reflex vertex p_i of D. If both edges incident to p_i are common edges of P and D then clearly both resulting dissection polygons have a common edge with the frame, namely incident to p_i.

Otherwise an edge incident to p_i must have been hit by a ray $\overrightarrow{a}$ that was shot from another reflex vertex b in the immediately preceding iteration. (Whenever an edge incident to a reflex vertex is hit, the next ray is shot from that reflex vertex.) Observe that both edges incident to b must be common edges of P and the corresponding dissection polygons because otherwise Drag-Edge would have been applied to b. Let $\overline{bc}$ denote the common edge of P and D. As the ray $\overrightarrow{s}$ is shot in direction of the edge that was hit by $\overrightarrow{a}$, $\overrightarrow{s}$ cannot hit back to $\overrightarrow{a}$. Moreover, $\overrightarrow{s}$ does not hit $\overline{bc}$ because then Drag-Edge would have been applied to p_i. Thus when D is dissected along $\overrightarrow{s}$ there are common edges with P in both resulting dissection polygons: On one side $\overline{bc}$, on the other side the edge incident to p_i that was not hit by $\overrightarrow{a}$. This proves (*D2*).

(*D5*) is again a consequence of the canonical dissection together with the fact that Extend-Reflex is called whenever a ray crosses any segment in the interior of the frame. □

Observe that at the end of Algorithm 3.18 all dissection polygons are convex and hence the dissection $\mathcal{D}$ is nice.

3.5 Simplification

In this section we discuss the simplification step in Phase 4. In order to construct a simple polygon from a frame P, we have to remove one occurrence of every double vertex from $P_{\circlearrowleft}$. Our strategy is to eliminate a reflex occurrence whose existence is guaranteed by Proposition 3.4.

Consider a reflex occurrence p_i in $P_{\circlearrowleft}$ of a double vertex. The obvious way to reduce p_i to a single vertex of P is to simply omit it, that is, to replace $(p_{i\ominus1}, p_i, p_{i\oplus1})$ by $(p_{i\ominus1}, p_{i\oplus1})$ in $P_{\circlearrowleft}$. This can be done if $(p_{i\ominus1}, p_i, p_{i\oplus1})$ forms what we call a *wedge* that is defined below.

Definition 3.21 *For $k \in \mathbb{N}$ consider a subpath $U = (p, q_1, \ldots, q_k, r)$ of $P_{\circlearrowleft}$ where q_i, $1 \le i \le k$, are reflex or flat vertices such that $C := (r, q_k, \ldots, q_1, p, r)$ forms a convex polygon.*

If $(\mathcal{R}(C) \setminus \partial U) \cap \partial S = \emptyset$ then the sequence $(q_1, \ldots, q_k)$ is called a **cap** *in $P_{\circlearrowleft}$. Otherwise $(q_1, \ldots, q_k)$ is called an* **anti-cap** *in $P_{\circlearrowleft}$.*

For $k = 1$ we usually omit the parentheses and call the vertex q_1 a cap or anti-cap.

A subpath $(p, q_1, \ldots, q_k, r)$ of $P_{\circlearrowleft}$ is called a **wedge** *if and only if q_i is a double vertex of P, for all $1 \le i \le k$, and $(q_1, \ldots, q_k)$ is a cap.*

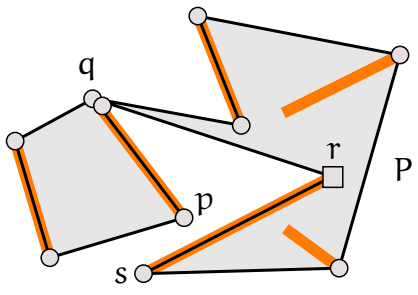


Figure 31: *The reflex occurrence of vertex q in $P_{\circlearrowleft}$ forms a cap and (p, q, r) is a wedge. But r is an anti-cap because $\overline{pq}$ intersects the triangle $\triangle(r, q, s)$.*

Observe that for C to form a convex polygon in Definition 3.21 one of

the vertices q_i, $1 \leq i \leq k$, must be reflex in $P_{\circlearrowleft}$. Figure 31 shows an example frame with both caps and anti-caps for illustration. If in a figure we want to emphasize that a reflex vertex is an anti-cap then it is shown by a square dot. If we can ensure that every double vertex of $P_{\circlearrowleft}$ appears at least once in a wedge, it is easy to create a simple polygon from a frame by means of the following operation.

Operation 5 (Chop-Wedges(P)) *(Figure 32)*

Input: a frame P.

Operation: As long as there is a wedge $(p, q_1, \ldots, q_k, r)$, replace the path $(p, q_1, \ldots, q_k, r)$ in $P_{\circlearrowleft}$ by the single edge (p, r).

Output: P.

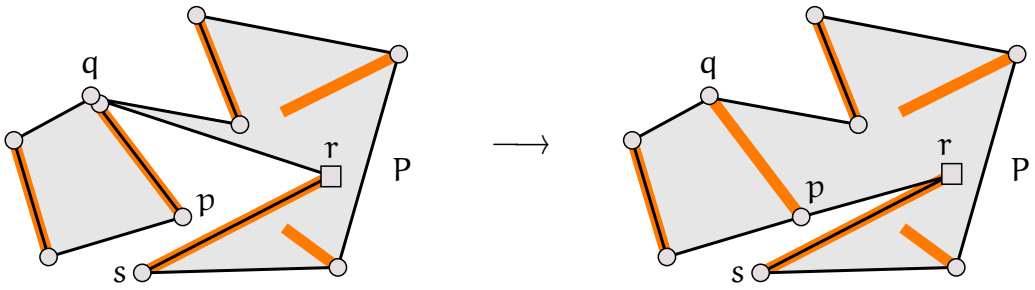


Figure 32: *Chopping the wedge* (p, q, r).

Observe that Chop-Wedges does not change $V(P)$. In fact, we will not chop off wedges as arbitrarily as the description above might suggest. Instead for each double vertex we will label one occurrence as a wedge, that is, as "to be removed" by Chop-Wedges. Obviously it has to be ensured that this particular occurrence is indeed part of a wedge in $P_{\circlearrowleft}$.

As for the other operations we like to prove that also the result of Chop-Wedges applied to a frame yields again a frame. The following proposition provides a first step in that direction.

Proposition 3.22 *The output of Chop-Wedges is a simply-connected polygon that satisfies (F1)–(F3).*

Proof. We have to show that chopping off a wedge $U = (p, q_1, \ldots, q_k, r)$ from $P_{\circlearrowleft}$ does not invalidate any of the frame properties mentioned. Denote the output polygon of Chop-Wedges by P'.

(*F1*) holds because $\mathcal{R}(P') \supset \mathcal{R}(P)$. Clearly no vertex appears more often in $P'_{\circlearrowleft}$ as it appears in $P_{\circlearrowleft}$. Therefore Property (*F3*) is satisfied as well.

Let $K := \mathcal{R}((p, q_1, \ldots, q_k, r, p)) \setminus \partial U$. By definition of wedge it is $K \cap \partial S = \emptyset$, in particular $\overline{pr}^{\ominus} \cap \partial S = \emptyset$ which implies (*F2*).

We claim that no edge incident to one of q_i in P, $1 \leq i \leq k$, can intersect K. By definition of wedge this is certainly true for segment edges, but there could be a visibility edge of P incident to one of these vertices. Suppose such an edge exists. If it is a cut-edge of P then $P_{\circlearrowleft}$ must traverse this cut-edge after q_i instead of proceeding along U. Else there is an edge incident to q_i and intersecting K that is not a cut-edge of P. But then P is not simply-connected, in contradiction to the assumption that P is a frame. Hence the claim holds and as a consequence P' is simply-connected. □

3.5.1 Labeling Wedges

As far as the inductive bridging step is concerned, the most important property of the dissection is (*D2*) which ensures a common edge of each dissection polygon $D \in \mathcal{D}$ with the surrounding frame. Along the line of Proposition 3.20 we roughly know how to maintain an almost nice dissection during the Saturation and Dissection Phase. But now we also have to take into account the Simplification Phase where some edges of the frame dissappear as they are part of a wedge that is chopped off in Chop-Wedges.

As already indicated when Chop-Wedges was introduced, we do not chop off wedges arbitrarily, but instead we *label* one occurrence of each double vertex in $P_{\circlearrowleft}$ as a wedge. A vertex can become a double vertex of the frame in three different ways, and for each of these cases we describe below which occurrence of the vertex is to be labeled as a wedge.

- Whenever a vertex is revisited on a geodesic, we label the original occurrence of the vertex as (part of) a wedge.

- In the case where Extend-Reflex generates many cut-edges (Figure 26 on Page 66) we label the vertex occurrences that are closer to t than to r as (part of) a wedge.

- Whenever Drag-Edge is applied to a reflex vertex p_i, we label the original occurrence of p_i as (part of) a wedge.

An edge in $P_{\circlearrowleft}$ is called a *wedge-edge* if and only if one of its endpoints is labeled as a wedge.

Obviously it has to be ensured that the labeled occurrence of a vertex indeed corresponds to a wedge as defined in Definition 3.21. This is our goal in the following. That means we must never label an anti-cap as a wedge or create an anti-cap at a vertex that is labeled as a wedge.

As a first step let us characterize certain classes of vertices that may never become a wedge during Algorithm 3.18.

Definition 3.23 *Consider a frame P with an almost nice dissection $\mathcal{D}$ of P. A vertex v in $P_{\circlearrowleft}$ is called* safe *if and only if*

- *v is strictly convex in $P_{\circlearrowleft}$;*
- *or v is endpoint of a segment diagonal;*
- *or v is a reflex single vertex of P that is incident to a dissection edge that is not an edge of P and that cuts $\measuredangle_P v$ into two strictly convex angles;*
- *or v is a double vertex of P that is not labeled as a wedge.*

A reflex twin $(a, \ldots, b)$ in $P_{\circlearrowleft}$ is safe *if and only if at least one of a or b is safe in $P_{\circlearrowleft}$. A vertex or reflex twin in $P_{\circlearrowleft}$ that is not safe is referred to as* unsafe.

As an example for the last type of safe vertices one may think of the second occurrence of a vertex generated by Drag-Edge, where the original occurrence was labeled as a wedge according to our labeling rules. The following proposition legitimates why those vertices are called safe.

Proposition 3.24 *A safe vertex of a frame is strictly convex in all dissection polygons where it appears. A safe vertex will never become unsafe or be labeled as a wedge in the course of Algorithm 3.18.*

Proof. During Algorithm 3.18 the frame P and its dissection $\mathcal{D}$ are modified by the three basic operations Build-Cap, Extend-Reflex, and Drag-Edge only, or one dissection polygon is simply split along a ray

emanating from one of its reflex vertices. Consider a safe vertex p_i in $P_{\circlearrowleft}$.

If p_i is strictly convex in $P_{\circlearrowleft}$ then no geodesic will revisit it. Also neither Extend-Reflex nor Drag-Edge can be applied to p_i. All basic operations decrease the angles at both vertices of the edge they replace. Hence p_i remains strictly convex throughout.

If p_i is an endpoint of a segment diagonal then it is a single vertex of P that is strictly convex in both dissection polygons where it appears. Therefore we can argue as for the strictly convex vertices above. Similarly, if p_i is a reflex single vertex of P that is incident to a dissection edge that is not an edge of P and that cuts $\measuredangle_P \nu$ into two strictly convex angles.

It remains to consider the case that ν is a double vertex of P that is not labeled as a wedge. Then by our labeling scheme the other occurrence of ν in $P_{\circlearrowleft}$ is labeled as a wedge. According to $(F4)$ ν appears as a strictly convex vertex in both dissection polygons and we can once again argue as before. □

As a consequence of Proposition 3.24 we do not have to worry about anti-caps created at safe vertices of the frame because they will never be labeled as a wedge during the algorithm. Hence our interest is focused at unsafe anti-caps in the following.

3.5.2 Anti-Cap Control

In this section we will analyze the basic operations and show that they do not create too many unsafe anti-caps. We cannot avoid that some unsafe anti-caps are created in the course of the algorithm, but we better make sure that they do not become double vertices of the frame. As for the dissections (nice versus almost nice) it turns out that we are fine if there exists at most one unsafe anti-cap at a time. The reason is that we can immediately deal with a single anti-cap by shooting the next ray from there.

We say that an operation *creates* an anti-cap if and only if in the output frame P' there appears an anti-cap that was not already an anti-cap in the input frame P. Observe that if an operation creates an anti-cap then this anti-cap appears locally, at one of the vertices for which an incident edge changes during the operation.

Proposition 3.25 *Consider a frame P with an orientation u that does not contain any alternation. Moreover, suppose that there is no unsaturated segment $s \in S$ whose endpoint in $V(P)$ is oriented towards an unsafe reflex vertex in $P_{\circlearrowleft}$. If Saturate is applied to (P, u) then it does not create any unsafe anti-cap and it does not generate any unsafe reflex twin.*

Proof. Consider a single operation Build-Cap and denote the vertices as in Proposition 3.6 by p_i, q, r, and $\mathrm{geo}(q, p_i, r) = (q = q_0, \ldots, q_k = r)$, for some $k \in \mathbb{N}$. Denote the input frame of this Build-Cap by Q and the output frame by Q'. Assume without loss of generality that $u(p_i) = +1$.

Build-Cap produces exactly one new reflex vertex: q. Let $C := (q_0, \ldots, q_\ell)$, where ℓ is chosen maximally such that $\ell < k$ and q_j is flat in ${Q'}_{\circlearrowleft}$, for all $1 \leq j \leq \ell$. Then C is a cap because the interior of the triangle $\triangle(p_i, q_{\ell+1}, q)$ and $\overline{p_i q_\ell}^{\ominus}$ are disjoint from ∂S by the definition of geodesic.

By (*F5*) p_i is a convex single vertex of Q and hence a strictly convex single vertex of Q'. Hence if p_i is part of a cap $(\ldots, t, p_i)$ in $Q_{\circlearrowleft}$ (implies that p_i is flat in $Q_{\circlearrowleft}$) then $(\ldots, t)$ is a cap in ${Q'}_{\circlearrowleft}$ that ends at t.

For the successor r of p_i in $Q_{\circlearrowleft}$ the situation is potentially different because its predecessor changes from p_i to q_{k-1}. Suppose that r is (part of) a cap $(r, v, \ldots)$ in $Q_{\circlearrowleft}$. (Otherwise there is nothing more to show.)

If r is flat in $Q_{\circlearrowleft}$ then r is strictly convex in ${Q'}_{\circlearrowleft}$ and $(v, \ldots)$ remains a cap in ${Q'}_{\circlearrowleft}$.

Otherwise r is reflex in $Q_{\circlearrowleft}$.

If r was already present in $P_{\circlearrowleft}$ then r is safe in $P_{\circlearrowleft}$: Suppose r is unsafe in $P_{\circlearrowleft}$. As p_i is oriented towards an unsafe reflex vertex in $Q_{\circlearrowleft}$, p_i cannot have been present in $P_{\circlearrowleft}$. Hence p_i was created in some previous Build-Cap operation during Saturate. But then the vertex to which Build-Cap was applied, must have been oriented towards r. It follows inductively that there is a vertex in $P_{\circlearrowleft}$ that is oriented towards r, in contradiction to the assumption that in $P_{\circlearrowleft}$ no vertex is oriented towards an unsafe reflex vertex. Hence if r was already present in $P_{\circlearrowleft}$ then r is safe in $P_{\circlearrowleft}$ and any anti-cap or reflex twin created at r is safe in ${Q'}_{\circlearrowleft}$ and hence by Proposition 3.24 in ${P'}_{\circlearrowleft}$.

Otherwise the vertex r has been created in this Saturate step in a preceding Build-Cap operation where a segment $\overline{rs}$ from S has been

saturated and $u(s) = -1$. But then s and its predecessor in the frame at that point form an alternation. As Build-Cap preserves orientations, that alternation must also exist in $P_{\circlearrowleft}$, in contradiction to our assumption that u does not have any alternation in $P_{\circlearrowleft}$. Therefore r appears already as a vertex in $P_{\circlearrowleft}$ and we can argue as above. □

Next, we would like to prove an analog to Proposition 3.25 for Extend-Reflex. Unfortunately Extend-Reflex can create unsafe anti-caps but — fortunately — at most one, as long as we can ensure that the vertex r that appears among the parameters is convex or safe in $P_{\circlearrowleft}$. (In the preconditions of Extend-Reflex r is required to be convex in the dissection polygon D, but this does not necessarily imply that it is a convex vertex in the frame.)

Proposition 3.26 *If the vertex r in Extend-Reflex is convex or safe in $P_{\circlearrowleft}$ then Extend-Reflex creates only one unsafe reflex twin, namely (q, t), and at most one unsafe anti-cap, at one of q or t.*

Proof. Consider an operation Extend-Reflex and denote the vertices as in the description of Extend-Reflex by p_i, r, q, and t, and let x be the point where the ray $\overrightarrow{s}$ hits $\overline{qt}$. Recall that $x \in \overline{qt}^{\ominus}$ as a precondition of Extend-Reflex, in particular $\overrightarrow{s}$ and $\overline{qt}$ are not collinear. As usual denote the input frame of the operation by P and the output frame by P'. Assume without loss of generality $r = p_{i\oplus 1}$. We have to verify that caps in $P_{\circlearrowleft}$ remain caps in $P'_{\circlearrowleft}$.

Clearly p_i is a strictly convex single vertex in P'. Hence a possible cap $(\ldots, p_{i\ominus 1}, p_i)$ in $P_{\circlearrowleft}$ remains a cap $(\ldots, p_{i\ominus 1})$ in $P'_{\circlearrowleft}$.

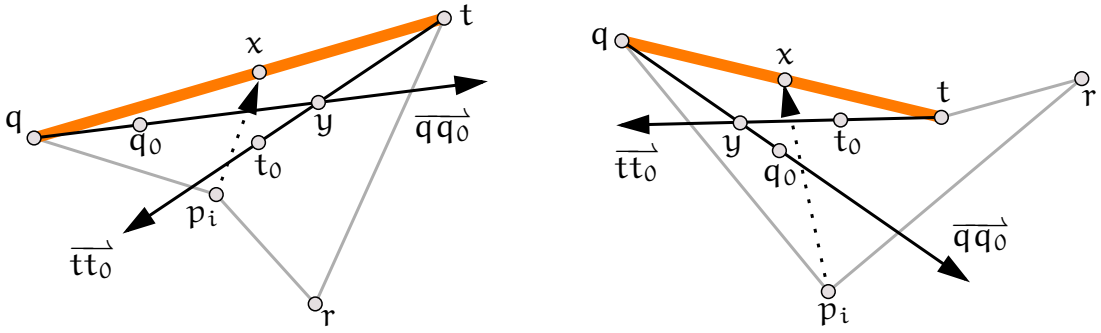
If r is convex in $P_{\circlearrowleft}$ then it is strictly convex in $P'_{\circlearrowleft}$. Hence a possible cap $(r = p_{i\oplus 1}, p_{i\oplus 2}, \ldots)$ remains a cap $(p_{i\oplus 2}, \ldots)$ in $P'_{\circlearrowleft}$. If r is reflex then it is safe in $P_{\circlearrowleft}$ by assumption and hence due to Proposition 3.24 also safe in $P'_{\circlearrowleft}$.

All interior vertices on the two geodesics are convex and "guarded" in $P'_{\circlearrowleft}$ by the safe vertices p_i or r. Compared to P there are two new reflex vertices in $P'_{\circlearrowleft}$: q and t. Clearly they form an unsafe reflex twin. We will show that at least one of them is a cap in $P'_{\circlearrowleft}$.

Let q_0 be the first strictly convex vertex of $\mathrm{geo}(q, x, p_i)$ (exists because p_i is strictly convex in $P'_{\circlearrowleft}$), and let t_0 be the first strictly convex vertex of $\mathrm{geo}(t, x, p_i, r)$ or r if no such vertex exists. Consider the rays $\overrightarrow{qq_0}$ and $\overrightarrow{tt_0}$ as shown in Figure 33. If these rays do not intersect within

the triangle $\triangle(p_i, x, q)$ then q is a cap because $\text{geo}(q, x, p_i)$ stays within this triangle and $\text{geo}(t, x, p_i, r)$ stays to the left of $\overrightarrow{tt_0}$ within the triangle $\triangle(p_i, t, x)$.

Hence suppose that the rays $\overrightarrow{qq_0}$ and $\overrightarrow{tt_0}$ intersect at a point y in the interior of $\triangle(p_i, x, q)$. Examine the segments $\overline{qq_0}$ and $\overline{tt_0}$: they cannot intersect since one stays within the triangle $\triangle(p_i, x, q)$ and the other stays within the quadrilateral $\square(t, x, p_i, r)$ and the intersection of both, the segment $\overline{p_i x}$, does not intersect ∂S in its relative interior. Hence it may be one of $y \in \overline{qq_0}$ or $y \in \overline{tt_0}$ but not both. If $y \in \overline{qq_0}$ then t is a cap (Figure 33(b)) else q is a cap (Figure 33(a)) in $P'_{\circlearrowleft}$. $\square$



(a) $\measuredangle(x, t, r)$ is convex. (b) $\measuredangle(x, t, r)$ is reflex.

Figure 33: *One of the segment endpoints* q *or* t *is a cap in Extend-Reflex.*

If t appears twice on $\text{geo}(t, x, p_i, r)$, we have to make sure that the reflex occurrence of t is a cap of $P'_{\circlearrowleft}$ that can be chopped off later. In general t can be an anti-cap, but the following corollary of Proposition 3.26 suggests a way to ensure that t is always a cap in that case.

Corollary 3.27 *If in Extend-Reflex* $\text{geo}(p_i, x, q) \subseteq \overline{p_i q}$ *and* t *appears twice on* $\text{geo}(t, x, p_i, r)$ *then the reflex occurrence of* t *in* $P'_{\circlearrowleft}$ *is a cap.*

Proof. If t is a reflex vertex of the quadrilateral $\square(p_i, r, t, x)$ then p_i is a convex vertex of $\square(p_i, r, t, x)$. Thus the ray $\overrightarrow{qq_0} = \overrightarrow{qp_i}$ in Proposition 3.26 cannot intersect $\overline{tt_0}$ in the interior of $\square(p_i, r, t, x)$ and hence the reflex occurrence of t is a cap in $P'_{\circlearrowleft}$. $\square$

If t appears twice on $\text{geo}(t, x, p_i, r)$ then the ray $\overrightarrow{s}$ can be slightly rotated around p_i towards q to form a ray $\overrightarrow{s_0}$ that still cuts $\measuredangle_D p_i$

into two strictly convex angles. More precisely, we rotate $\overrightarrow{s}$ around p_i towards q until it hits one of

q: (Figure 34(a)) then $\text{geo}(p_i, x, q) \subseteq \overline{p_i q}$ and t is a cap by Corollary 3.27;

t': that is, the right endpoint of another segment $\overline{q't'}$ that lies in D° (Figure 34(b)); then extend to the segment $\overline{q't'}$ instead; the quadrilateral in Extend-Reflex degenerates to the triangle $\triangle(p_i, r, t')$ and t' cannot appear twice on the corresponding geodesic;

p_k: where p_k is another reflex vertex of D (Figure 34(c)); in this case we do not apply Extend-Reflex.

Strictly speaking the above conditions are not well-defined since Extend-Reflex forbids $\overrightarrow{s}$ to hit a segment endpoint. But obviously we may suppose instead that the ray $\overrightarrow{s}$ hits a point on the segment that has a sufficiently small but positive distance to the mentioned endpoint.

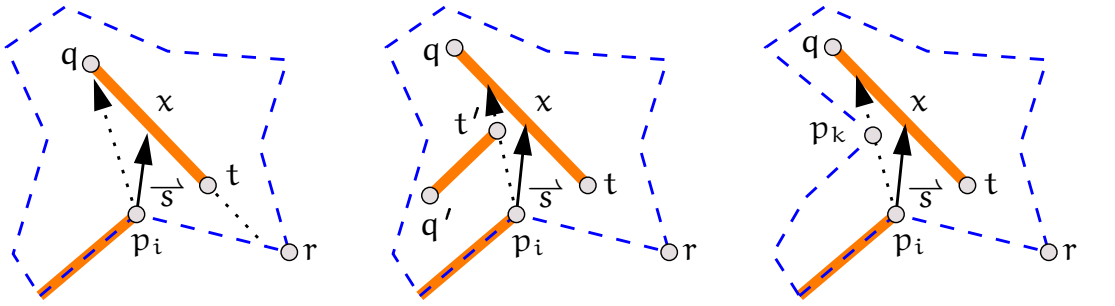


Figure 34: *The three possible results of the rotation of* $\overrightarrow{s}$ *around* p_i *towards* q.

Proposition 3.28 *After the rotation described above the ray* $\overrightarrow{s}$ *still cuts* $\measuredangle_D p_i$ *into two strictly convex angles.*

Proof. By Proposition 3.14 p_i is a common reflex single vertex of D and P. Let w and r denote the vertices of D adjacent to p_i such that without loss of generality $\measuredangle(w, p_i, r)$ is strictly convex. Before the rotation, $\overrightarrow{s}$ is to the left of $\overrightarrow{wp_i}$ and to the right of $\overrightarrow{rp_i}$.

If t appears twice on $\text{geo}(t, x, p_i, r)$ in Extend-Reflex then t is a reflex vertex of the quadrilateral $\square(t, x, p_i, r)$. This means that q is to

the right of $\overrightarrow{rp_i}$. As the rotation stops at q, the rotated ray still cuts $\measuredangle_D p_i$ into two strictly convex angles. □

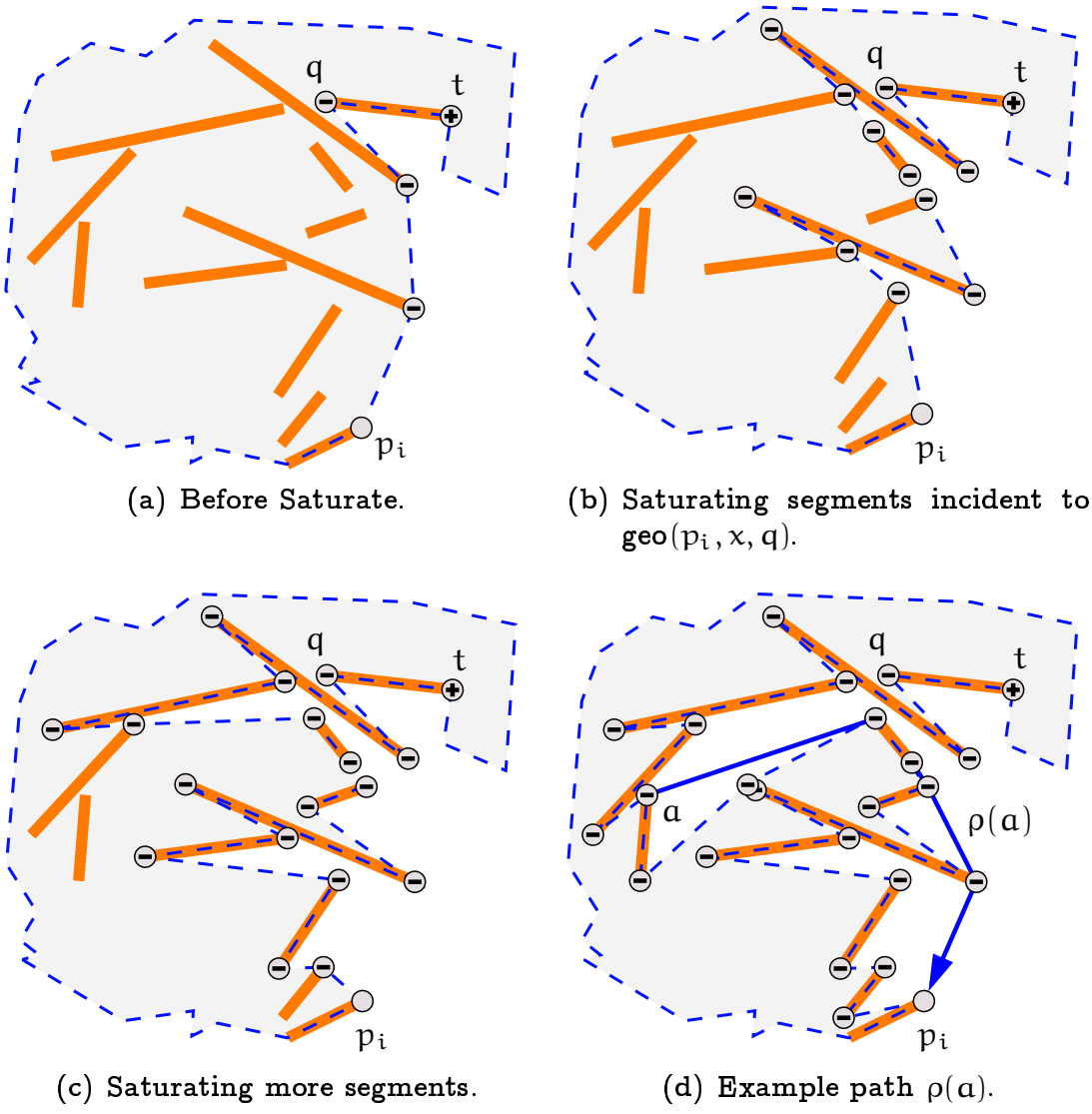
As Extend-Reflex possibly creates an unsafe anti-cap, we have to make sure that this anti-cap does not appear on a geodesic in one of the following steps. Partly this can be taken care of by shooting the next ray from the anti-cap. But before the next ray is shot another saturation step takes place.

Proposition 3.29 *Consider the two endpoints q and t of the interior segment in Extend-Reflex. Denote by P the input and by P' the output frame of Extend-Reflex. Let Q denote the output frame of the following call to Saturate. Then Saturate does not revisit q, that is, q appears as often in $Q_{\circlearrowleft}$ as it appears in $P'_{\circlearrowleft}$.*

Proof. All unsaturated segments in P' are incident to either $\text{geo}(p_i, x, q)$ or $\text{geo}(r, p_i, x, t)$. Consider only the segments incident to $\text{geo}(p_i, x, q)$ for the moment. Saturating them might introduce new unsaturated segments along the geodesics constructed by Build-Cap. These are then saturated as well and this process continues until none of the geodesics catches a new segment. Let a be a vertex that is visited by Build-Cap at some point during this procedure. Recall that Build-Cap retains the orientation and propagates it all along the constructed geodesic. Hence it is $u(a) = -1$.

Define a path $\rho(a)$ from a to p_i as follows: go from a along the first geodesic that visited a in this Saturate step in direction of the convex endpoint c of the geodesic (that is, away from the endpoint of the segment that has been saturated in this particular Build-Cap operation). Then from c proceed along $\rho(c)$. The base case for this recursive definition is provided by the interior vertices of $\text{geo}(q, x, p_i)$ from where we simply walk along $\text{geo}(q, x, p_i)$ to p_i. Figure 35(d) shows an example for a vertex a and its corresponding path $\rho(a)$.

As argued above it is $u(v) = -1$ for all $v \in V(\rho(a))$. This implies that $\rho(a) = (a = a_1, \ldots, a_\ell = p_i)$, $\ell \in \mathbb{N}$, is a right-turn, that is, $\measuredangle(a_{k+1}, a_k, a_{k-1})$ is reflex or flat, for all $1 < k < \ell$. Furthermore, $\rho(a)$ is non-crossing because every edge of a constructed geodesic becomes an edge of the frame immediately that will not be crossed by any further geodesic according to Proposition 3.6. Finally observe that $\partial(\rho(a)) \subset \mathcal{R}(P')$, $\partial(\rho(a)) \cap Q^\circ = \emptyset$, and at each interior vertex the incident



(a) Before Saturate.

(b) Saturating segments incident to $\mathrm{geo}(p_i, x, q)$.

(c) Saturating more segments.

(d) Example path $\rho(a)$.

Figure 35: *Saturate does not revisit* q.

segment from S lies within the convex angle, that is, to the right of (or on) $\rho(a)$.

Suppose that q is visited by Saturate and let b be the endpoint of the segment that is saturated in this operation such that b is not a vertex of the frame at that point. (Recall that the segment incident to q, $\overline{qt}$, is already part of the frame before the call to Saturate.) Then the path $\rho(b)$ induces a non-crossing right-turn path R within $\mathcal{R}(P')$ from q to p_i such that q is a reflex or flat vertex of the path $R' := \mathrm{geo}(p_i, x, q) \cdot R$ and $\overline{qt}$ is to the right of (or on) R'. But as the convex angle at q is exterior to P', the path R' describes a closed curve within P' which is not null-homotopic, in contradiction to P' as a frame being simply-connected.

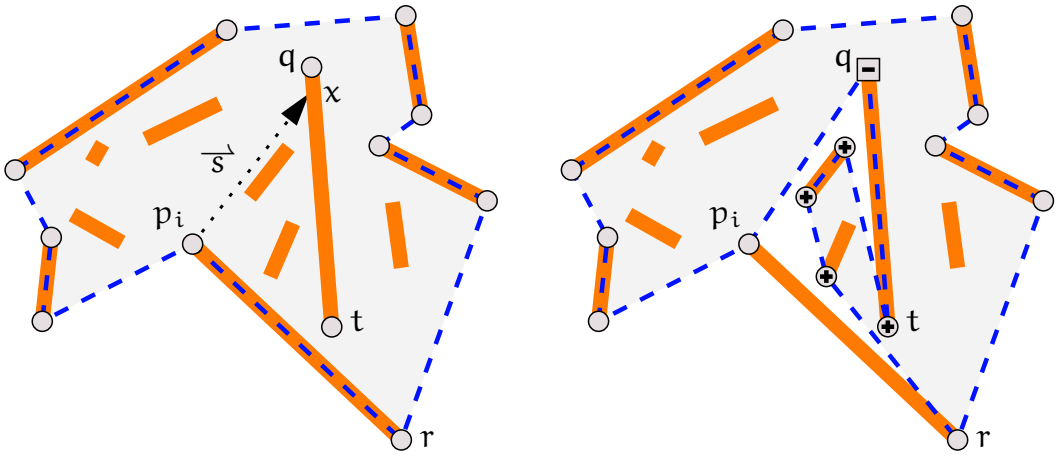
The argument for the vertices visited by a Build-Cap operation with positive orientation, that is, initiated by some unsaturated segment incident to $\mathrm{geo}(r, p_i, x, t)$, is symmetric. For a vertex a visited by such an operation we can define a path $\sigma(a)$ similar to $\rho(a)$ that connects a to r. The path $\sigma(a)$ is a left-turn path within $\mathcal{R}(P')$ and at each interior vertex the incident segment from S lies within the convex angle, that is, to the left of (or on) $\sigma(a)$. In the same way as above we get a contradiction to the fact that P' is simply-connected if such a Build-Cap operation revisits q.

In summary we conclude that Saturate does not revisit q. □

In contrast to q the the other segment endpoint t may be revisited during the call to Saturate directly after the Extend-Reflex operation as the example shown in Figure 36 demonstrates. But this can happen only if t is a reflex vertex of the quadrilateral $\square(t, x, p_i, r)$. If t is a convex vertex of $\square(t, x, p_i, r)$, we can argue as in Proposition 3.29 to get the following statement.

Corollary 3.30 *Consider the two endpoints q and t of the interior segment in Extend-Reflex and suppose that t is a convex vertex of the quadrilateral $\square(t, x, p_i, r)$. Denote by P the input and by P' the output frame of Extend-Reflex. Let Q denote the output frame of the following call to Saturate. Then Saturate does not revisit t, that is, t appears as often in $Q_{\circlearrowleft}$ as it appears in $P'_{\circlearrowleft}$.* □

The corollary below summarizes the discussion about unsafe anti-caps potentially created by Extend-Reflex.



(a) Before Extend-Reflex. (b) Before Saturate.

Figure 36: *Vertex* t *may be revisited during Saturate.*

Corollary 3.31 *If Extend-Reflex creates an unsafe anti-cap then this anti-cap is not revisited during the following call to Saturate.*

Proof. As discussed in Proposition 3.26, Extend-Reflex can create an unsafe anti-cap at one of q or t only. If it creates an anti-cap at q then q is not revisited in the following Saturate step according to Proposition 3.29. If Extend-Reflex creates an anti-cap at t then either t is a convex vertex of the quadrilateral $\square(t, x, p_i, r)$ and the claim follows by Corollary 3.30, or t is a reflex vertex of $\square(t, x, p_i, r)$ and then due to the rotation of $\overrightarrow{s}$ described in Proposition 3.28 t is a cap in the resulting frame by Corollary 3.27. □

Let us conclude the analysis of anti-cap creations by looking at the last remaining operation: Drag-Edge.

Proposition 3.32 *Consider an operation Drag-Edge and its parameters* P, p_k *and* p_ℓ. *If* p_ℓ *is not part of an unsafe reflex twin in* $P_\circlearrowleft$ *and* p_k *is convex or safe in* $P_\circlearrowleft$ *then Drag-Edge does not create any unsafe reflex twin and it creates at most one unsafe anti-cap: at* p_ℓ.

Proof. Denote the vertex from which the ray $\overrightarrow{s}$ is shot by p_i, as in the description of Drag-Edge. Recall that $\overrightarrow{s}$ hits $\overline{p_k p_\ell}^{\ominus}$ as a precondition of Drag-Edge, in particular $\overrightarrow{s}$ and $\overline{p_k p_\ell}$ are not collinear. As usual denote the output frame of Drag-Edge by P'. As a precondition of Drag-Edge

the segment $\overline{p_k p_\ell}$ is an edge of P, assume without loss of generality $p_\ell = p_{k\oplus 1}$. We have to verify that caps in $P_\circlearrowleft$ remain caps in $P'_\circlearrowleft$.

The occurrence of p_i in $P_\circlearrowleft$ is not modified and thus remains a cap in $P'_\circlearrowleft$ if it was (part of) a cap in $P_\circlearrowleft$. Compared to $P_\circlearrowleft$ the vertex p_i appears a second time in $P'_\circlearrowleft$: as a common endpoint of both geodesics constructed in Drag-Edge. By definition of geodesic this second occurrence is a cap. Moreover, according to our wedge labeling scheme the original occurrence of p_i is labeled as a wedge and hence the second occurrence is safe in $P'_\circlearrowleft$.

If p_k is convex in $P_\circlearrowleft$ then it is strictly convex in $P'_\circlearrowleft$. Hence a possible cap $(\ldots, p_{k\ominus 1}, p_k)$ remains a cap $(\ldots, p_{k\ominus 1})$ in $P'_\circlearrowleft$. Else p_k is safe in $P_\circlearrowleft$ by assumption and hence it is also safe in $P'_\circlearrowleft$ according to Proposition 3.24.

If p_ℓ is convex in $P_\circlearrowleft$ then we argue as above for p_k. Else p_ℓ is reflex in $P_\circlearrowleft$ and may become an unsafe anti-cap in $P'_\circlearrowleft$. It is not part of an unsafe reflex twin in $P'_\circlearrowleft$: on one side, all vertices along the constructed geodesic are convex and the other endpoint of the geodesic, p_i, is safe. On the other side, the successor of p_ℓ did not change from $P_\circlearrowleft$ to $P'_\circlearrowleft$. As p_ℓ was not part of an unsafe reflex twin in $P_\circlearrowleft$, it is thus neither part of an unsafe reflex twin in $P'_\circlearrowleft$.

All interior vertices on the two geodesics are convex and "guarded" in $P'_\circlearrowleft$ by the safe occurrence of p_i and either the safe vertex p_k or the unsafe vertex p_ℓ. □

As discussed in Proposition 3.32 the second (new) reflex occurrence of p_i in $P'_\circlearrowleft$, let us refer to it as p_m, is a safe cap. The orientation along the geodesics is chosen in such a way that there may be vertices incident to unsaturated segments that are oriented towards p_m. As a consequence p_m may become an anti-cap (as shown in Figure 37) or even a convex vertex of $P'_\circlearrowleft$ during the Saturate step following Drag-Edge. Therefore it is rather important that p_m is indeed safe in $P'_\circlearrowleft$.

As a consequence we must not apply Drag-Edge when p_i is an anti-cap in the frame. For the (possibly) unsafe anti-cap p_ℓ we can argue as for Extend-Reflex in Proposition 3.29 that p_ℓ is not revisited during the following call to Saturate.

Corollary 3.33 *Consider the vertex p_ℓ in Drag-Edge. Denote by P the input and by P' the output frame of Drag-Edge. Let Q denote the*

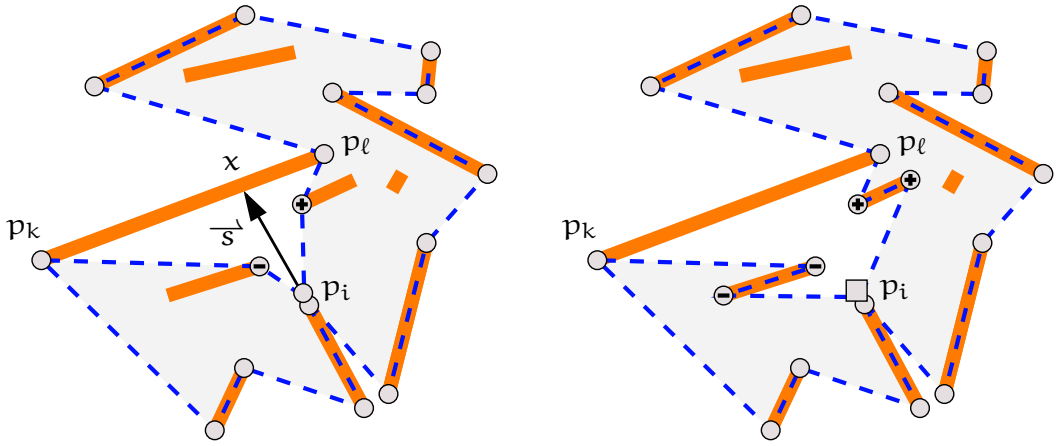


Figure 37: *The Saturate step after Drag-Edge may create an anti-cap at* $p_i = p_m$.

output frame of the following call to Saturate. Then Saturate does not revisit p_ℓ*, that is,* p_ℓ *appears as often in* $Q_\circlearrowleft$ *as it appears in* $P'_\circlearrowleft$. □

3.6 Preparations for Bridging

Before we summarize the algorithm for the Saturation, Dissection, and Simplification Phases and analyze the resulting dissections, we have to add one final bit in anticipation of the inductive bridging step that follows. As often in inductive proofs we prove a slightly stronger statement that enables us to put together the inductively computed solutions to different sub-problems: We want to fix an arbitrary edge $\{y, z\}$ of the initial frame $P = \text{conv}(\partial S)$ that must stay an edge of the frame all through the algorithm, in particular, it is an edge of the Hamiltonian polygon to be constructed.

Assume without loss of generality that z follows y in $P_\circlearrowleft$. If $\{y, z\}$ is a visibility edge and the segment from S incident to y is unsaturated by P then the edge $\{y, z\}$ will be replaced by a geodesic during the Saturation Phase already. Hence we have to slightly adapt the orientation of P by setting

$$u_y(v) := \begin{cases} -1 & , v = y, \\ +1 & , \text{otherwise.} \end{cases}$$

This small change compared to the uniform orientation is already suffi-

cient to ensure that $\{y, z\}$ remains an edge of the frame throughout the algorithm.

Proposition 3.34 *If the initial frame is oriented by u_y and if Drag-Edge is not applied when a ray hits the edge $\{y, z\}$ then $\{y, z\}$ is an edge of the frame all through the Saturation, Dissection, and Simplification Phases.*

Proof. The frame is modified using the basic operations only. Hence it is sufficient to prove that none of these operations replaces $\{y, z\}$. As convex hull vertices both y and z are convex vertices in every frame for S. We will argue that none of the basic operations creates a reflex vertex at y or z.

Build-Cap creates a reflex vertex only at an endpoint q of an unsaturated segment where $q \in P^\circ$. Extend-Reflex creates reflex vertices only at endpoints of a segment $s \subset P^\circ$. Drag-Edge does not create any new reflex vertices but just a new occurrence of an existing reflex vertex. As convex hull vertices both y and z cannot appear as interior vertex on one of the geodesics, either. Thus both y and z remain convex single vertices of the frame throughout the Saturation and Dissection Phases.

Observe that Extend-Reflex only replaces edges that are incident to a reflex vertex. Drag-Edge does not replace $\overline{yz}$ by assumption. Chop-Wedges only replaces edges incident to a double vertex of the frame. Finally, Build-Cap only replaces edges for which one endpoint is oriented towards the other. As the orientation of y and z has been chosen such that both are oriented away from each other, Build-Cap does not replace $\{y, z\}$, either, and the claim follows. □

Changing to a non-uniform orientation has consequences for some of the claims that have been made in the preceding sections. In particular we cannot use Proposition 3.25 to claim that the initial call to Saturate does not create any unsafe anti-cap. Indeed, the simple example given in Figure 38 shows that Saturate may create an unsafe anti-cap if the orientation of the frame contains an alternation. But as the orientation u_y is almost uniform, the impact is not dramatic. We will discuss the consequences in the next section.

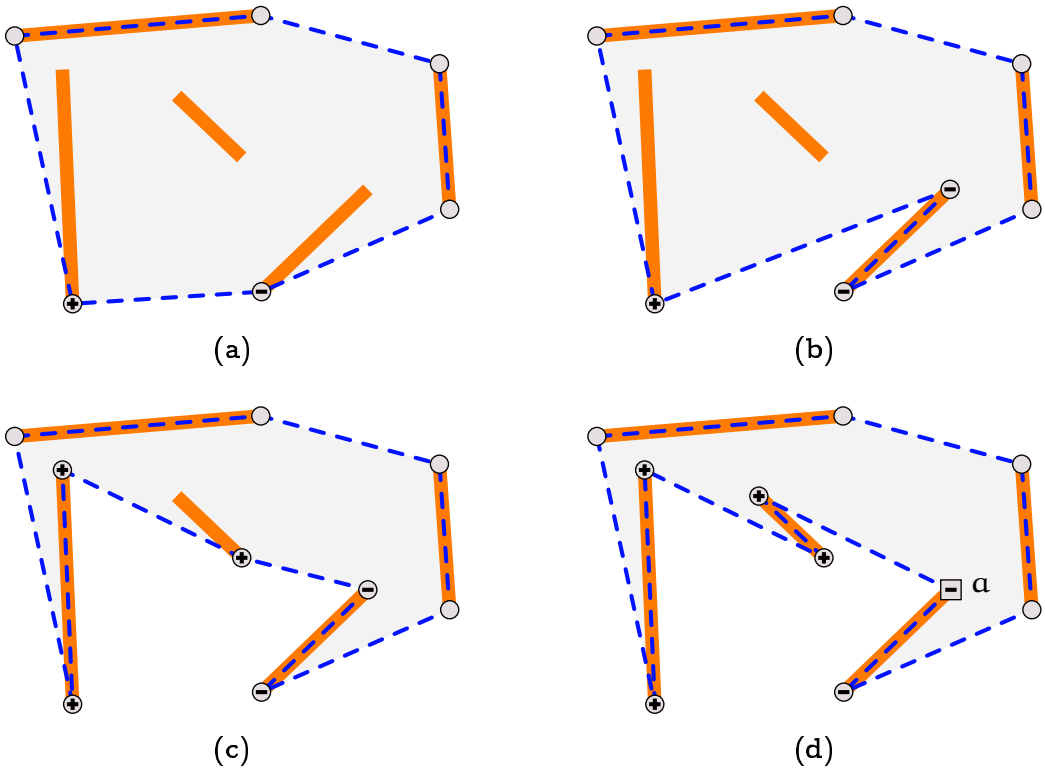


Figure 38: *Alternations in the orientation may lead to an unsafe anti-cap (vertex* a *in Figure 38(d)) in Saturate.*

3.6.1 Wedge Control

Recall that in order to maintain an (almost) nice dissection we need to ensure a common edge of each dissection polygon $D \in \mathcal{D}$ with the surrounding frame. Compared to Algorithm 3.18 there are now two new complications: first, the future disappearance of wedges should be taken into account; and second, the special edge $\{y, z\}$ should not be counted as a common edge of the dissection polygon it bounds and the frame. Thus, we raise the requirements for the dissection to be maintained correspondingly.

Definition 3.35 *Consider a frame P for a set S of disjoint segments and a fixed edge $\{y, z\}$ of $\mathrm{conv}(\partial S)$. An almost nice dissection $\mathcal{D}$ of P is called* **beautiful** *if and only if*

- *(D2*) every polygon $D \in \mathcal{D}$ has a common edge with P that is not a wedge-edge nor it is the special edge $\{y, z\}$; such an edge is called a* **good edge** *for D in the following;*
- *(D3*) there is at most one unsafe reflex twin and at most one unsafe anti-cap in $P_{\circlearrowleft}$; if both exist then the anti-cap is part of the reflex twin;*
- *(D4*) for every reflex vertex r of some $D \in \mathcal{D}$ there is an incident edge $\overline{rq}$ that is good for D and such that q is a convex or safe vertex in $P_{\circlearrowleft}$; such an edge is called a* **perfect edge** *in the following.*

Notice that (*D2**) implies (*D2*), (*D3**) implies (*D3-*), and (*D4**) implies (*D4*).

Our goal is to maintain a beautiful dissection of the frame during the algorithm. However, already when Saturate is applied to the frame $\mathrm{conv}(\partial S)$ there might be many wedge-edges in the resulting frame. If Saturate generates a reflex twin at the two alternation vertices and one of the vertices of this reflex twin is revisited by a geodesic then one edge incident to the second reflex vertex p_i of the reflex twin is a wedge-edge. Even if the very first ray is shot from p_i, it may create a dissection polygon that has no good edge, as the example shown in Figure 39 demonstrates.

Moreover, as we have seen in Figure 38, Saturate may create an unsafe anti-cap and we have to ensure that this unsafe anti-cap is not

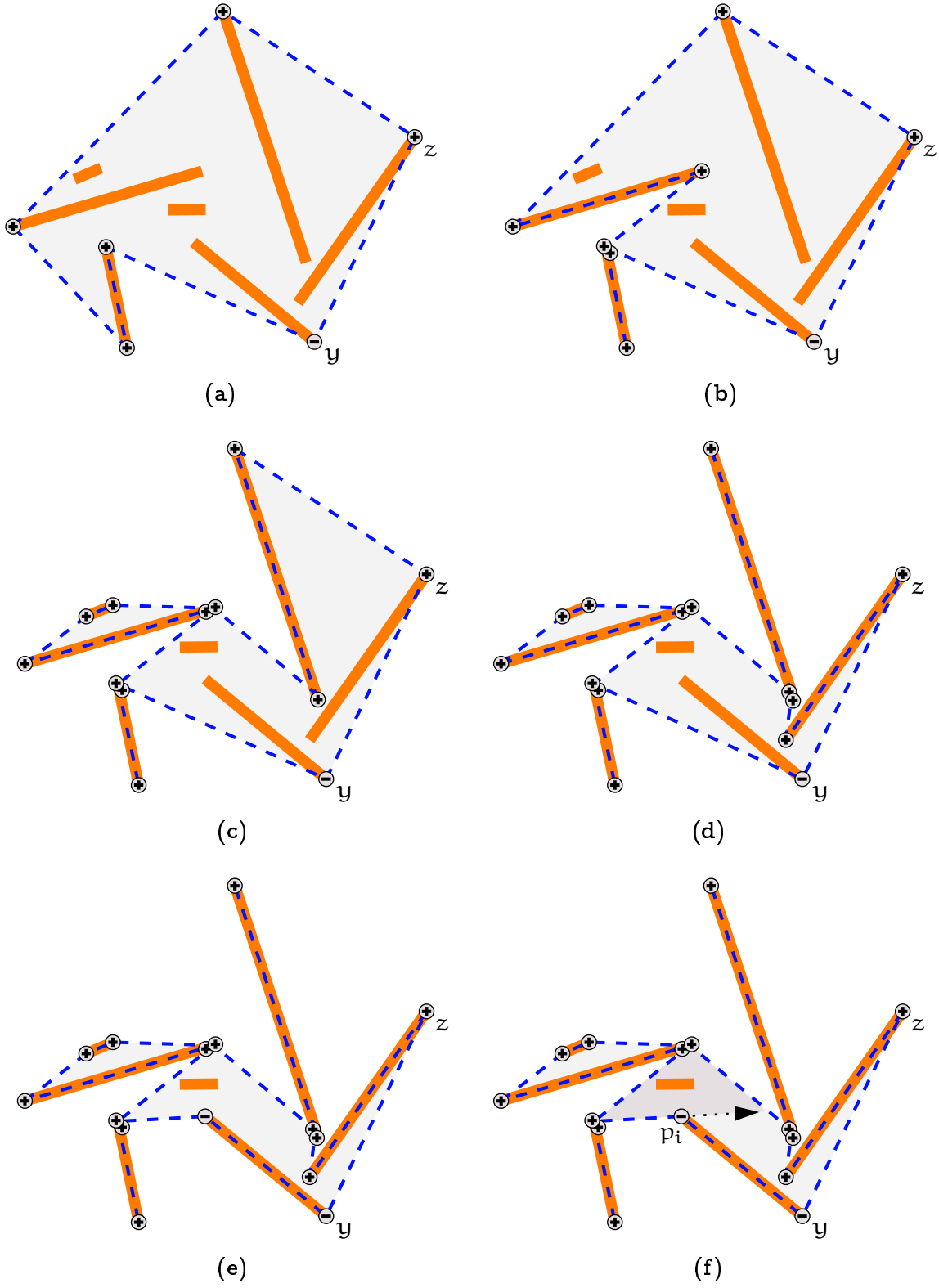


Figure 39: *Saturating segments in arbitrary order may lead to a dissection polygon that has no good edge: all common edges of the frame and the shaded region in Figure 39(f) are wedge-edges.*

revisited by a geodesic. Observe that these problems can occur only during the very first call to Saturate, as only at that point there is an alternation in the frame orientation.

The solution is rather straightforward: so far we made no assumption on the order in which unsaturated segments are saturated during Saturate. This changes now: We demand that those unsaturated segments become saturated last whose vertices in the frame

- participate in an alternation of the orientation (and thus by Proposition 3.10 potentially generate a reflex twin)
- or are oriented towards a reflex vertex of the frame.

Moreover, if one of the vertices that potentially becomes part of a reflex twin is revisited, we change the orientation for the remaining unsaturated vertices towards the convex re-occurrence of that vertex. In such a way we can simply avoid creating a reflex twin in this case. If, on the other hand, this vertex is not revisited until the very end then we do not mind creating an anti-cap there; as a single vertex of the frame we know how to deal with it.

The above discussion is summarized below in a refined variation of Saturate that we call *Careful-Saturate*. Clearly all claims that have been made about Saturate in the preceding sections also hold for Careful-Saturate because it just changes the order in which Build-Cap is applied to unsaturated vertices and the possible re-orientation does not induce any alternation or orientations towards a reflex vertex.

Operation 6 (Careful-Saturate(P, u))

Input: a frame P and an orientation $u(P) = u_y(P)$, for some $y \in V(P)$.

Operation:

As long as there is an unsaturated vertex in $V(P) \setminus \{y\}$, select p_i as the first unsaturated vertex that appears after (not including) y in $P_{\circlearrowleft}$ and let $(P, u) \leftarrow$ Build-Cap(P, u, p_i).

Let x denote the predecessor of y in $P_{\circlearrowleft}$.

As long as there is an unsaturated vertex in $V(P)$,

- select p_i as the last unsaturated vertex that appears before x in $P_{\circlearrowleft}$ and let $(P, u) \leftarrow$ Build-Cap(P, u, p_i).

- If the preceding Build-Cap operation revisited x then set $u(v) := +1$, for all v that appear between the original and the new occurrence of x in $P_{\circlearrowleft}$.

Output: (P, u).

Figure 40 illustrates Careful-Saturate running on a small example. We have to prove that Careful-Saturate indeed guarantees good edges incident to all reflex vertices, that not too many anti-caps are created, and that no anti-cap is a double vertex in the resulting frame.

Proposition 3.36 *Consider the frame $P = \mathrm{conv}(\partial S)$ with an edge $\{y, z\}$ of $P_{\circlearrowleft}$ and the orientation u_y. Apply Careful-Saturate to P and denote the resulting frame by P'. Then the following statements hold.*

- *There is at most one reflex twin in $P'_{\circlearrowleft}$ and at most one anti-cap.*
- *An anti-cap in $P'_{\circlearrowleft}$ can appear only at a reflex vertex that is part of a reflex twin. (More precisely, if a is an anti-cap in $P'_{\circlearrowleft}$ then there is a reflex twin $(a, \ldots, b)$.)*
- *For every reflex single vertex of P' both incident edges are good for the corresponding dissection polygon.*
- *If $(a, \ldots, b)$ forms a reflex twin in $P'_{\circlearrowleft}$ then both a and b are single vertices of P'.*

Proof. Let us first argue that both edges incident to a reflex single vertex of P' that is not part of a reflex twin are good for the corresponding dissection polygon.

Consider a reflex single vertex v in P' that is not part of a reflex twin. As a single vertex it is not part of a wedge by definition. Since v is not part of a reflex twin both neighbors are convex vertices in $P'_{\circlearrowleft}$. As no geodesic can revisit a flat vertex without also revisiting an adjacent (directly or via other flat vertices) reflex vertex, the neighbors of v are single vertices in $P'_{\circlearrowleft}$ and hence not part of a wedge, either. Finally, by Proposition 3.34 both y and z remain convex single vertices of the frame throughout and thus $\overline{yz}$ is not incident to any reflex vertex during Careful-Saturate, in particular not to v. Thus both edges incident to v are good for the corresponding dissection polygon, as claimed.

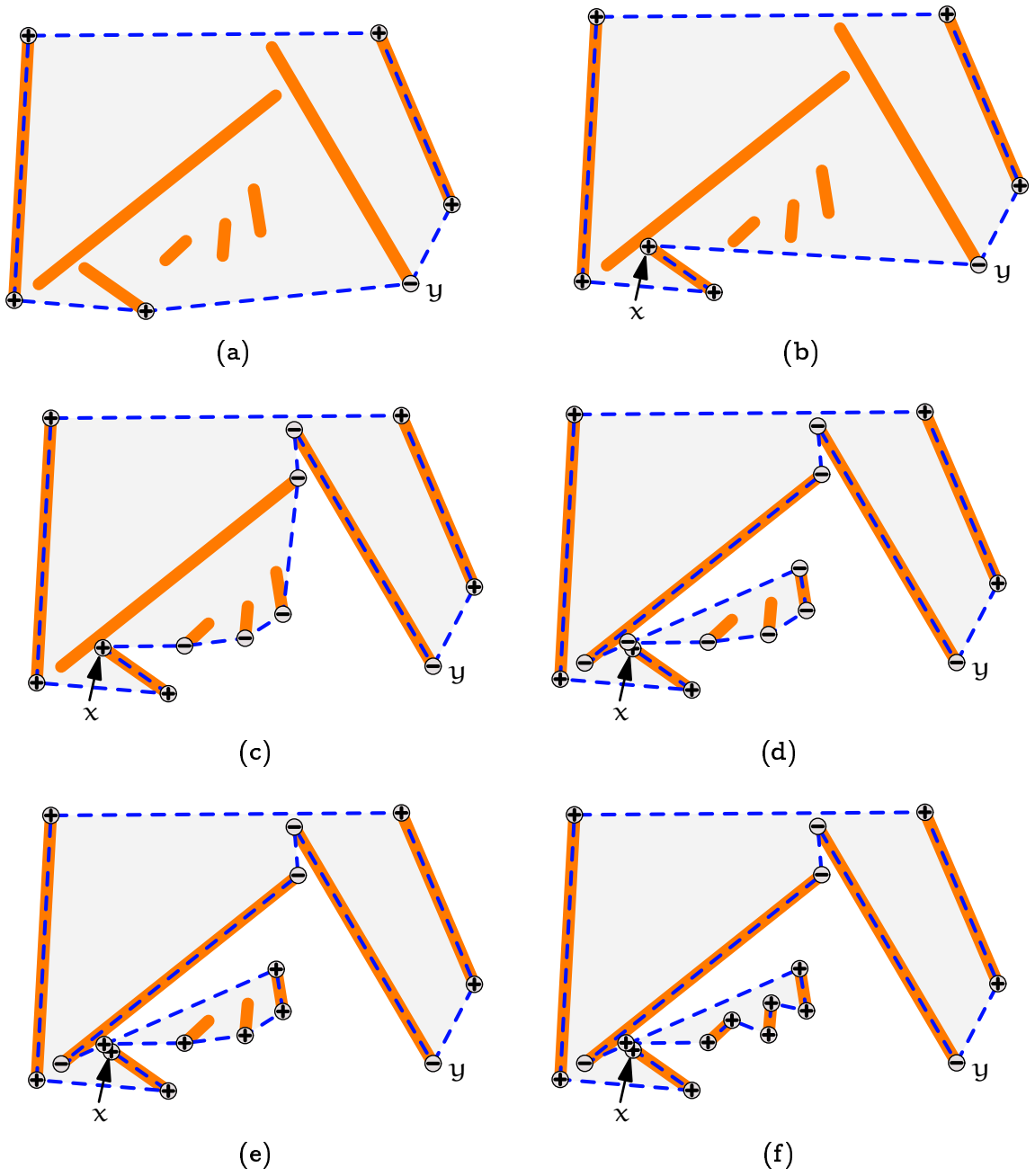


Figure 40: *Careful-Saturate: the re-orientation avoids creating an anti-cap at a wedge and guarantees good edges at reflex vertices.*

Certainly no vertex in P is oriented towards a reflex vertex in $P_{\circlearrowleft}$ because there are no reflex vertices in $\text{conv}(\partial S)$. Also, there is only one alternation, namely at y. Thus, by Proposition 3.10 there is at most one reflex twin in $P'_{\circlearrowleft}$. More precisely, as in Proposition 3.10 and Proposition 3.25, no reflex twin and no anti-cap is created before Build-Cap is applied to y in the second loop of Careful-Saturate. It might be that the segment $\overline{yy_0}$ from S incident to y is already saturated at that point. In this case there is nothing more to show. Therefore, suppose that Build-Cap is applied to y at begin of the second loop in Careful-Saturate.

If x is a convex vertex of the frame at the point where Build-Cap is applied to y then x is a strictly convex vertex of the resulting frame. Then no reflex twin will be created and we can once again argue as before. Thus suppose that x is a reflex vertex of the frame before and after Build-Cap is applied to y. By the order in which vertices were saturated during the first loop in Careful-Saturate, x is the last reflex vertex created in the first loop of Careful-Saturate; in particular, x is a single vertex of the frame at that point.

If $\text{geo}(y_0, y, x) \subseteq \overline{y_0 x}$ and there are no unsaturated vertices along $\text{geo}(y_0, y, x)$ then y_0 and x may form a reflex twin in the resulting frame. In this case both are single vertices of the frame and y_0 is a cap. As there are no more unsaturated segments, Careful-Saturate terminates immediately and there is nothing more to show.

Else if there is at least one strictly convex vertex but no unsaturated vertices along $\text{geo}(y_0, y, x)$ then no reflex twin is created and there is nothing more to show.

Otherwise there are some unsaturated vertices along $\text{geo}(y_0, y, x)$. The second loop of Careful-Saturate is designed to saturate these vertices in an order such that every time before Build-Cap is applied to a vertex p_i

- all unsaturated vertices are within the subwalk $(x, \ldots, p_i)$ of $P_{\circlearrowleft}$
- and x is the only reflex vertex in the subwalk $(x, \ldots, p_i)$ of $P_{\circlearrowleft}$.

Suppose that in some Build-Cap operation the geodesic revisits x and the vertices between the original and the new occurrence of x in $P_{\circlearrowleft}$, let us call it x', are re-oriented. As x is the only reflex vertex in the subwalk $(x, \ldots, x')$ of $P_{\circlearrowleft}$, no vertex is oriented towards a reflex

vertex in the frame after this re-orientation. Moreover, there is no alternation in the orientation because both x and x' are saturated, all unsaturated vertices within $(x, \ldots, x')$ are oriented uniformly $+1$, and all unsaturated vertices within $(x', \ldots, x)$ are oriented uniformly -1. Therefore we may conclude that the remaining iterations do not create any reflex twin or anti-cap in the same way as in Proposition 3.10 and Proposition 3.25.

It remains to consider the case that no Build-Cap operation during the second loop in Careful-Saturate revisits x. Then x is a single vertex of P'. Only the very last Build-Cap operation in Careful-Saturate may create a reflex twin, namely at x and the endpoint p_0 of the segment $\overline{p_i p_0}$ that is saturated in that last Build-Cap operation. Clearly p_0 is a single vertex of P' that is a cap and as above all edges incident to p_0 or x are good for their common dissection polygon. □

3.7 Algorithm Summary

This section describes the final algorithm for the Saturation, Dissection, and Simplification Phases and proves that it maintains a frame P together with a beautiful dissection $\mathcal{D}$ of P. Essentially, it is a refinement of Algorithm 3.18 adapted to the needs of the Simplification Phase.

Algorithm 3.37

Input: a set S of disjoint line segments and an edge $\{y, z\}$ of $\mathrm{conv}(\partial S)$.

Initialization:

P	$\leftarrow$	$\mathrm{conv}(\partial S)$.	*(frame)*
u	$\leftarrow$	u_y.	*(orientation)*
$\mathcal{D}$	$\leftarrow$	$\{P\}$.	*(dissection)*
D	$\leftarrow$	$\emptyset$.	*(dissection polygon)*
p_i	$\leftarrow$	$\emptyset$.	*(reflex vertex of D)*
r	$\leftarrow$	$\emptyset$.	*(neighbor of p_i in D and $P_{\circlearrowleft}$)*
w	$\leftarrow$	$\emptyset$.	*(the other ($\neq r$) neighbor of p_i in D)*

Algorithm:

Careful-Saturate(P, u) and dissect $\mathcal{D}$ canonically.

If Careful-Saturate created a reflex twin $(a_0, \ldots, a_k)$, $k \in \mathbb{N}$, in $P_{\circlearrowleft}$ then let $(p_i, r, w) \leftarrow (a_0, b, a_1)$, where b is the other $(\neq a_1)$ neighbor of a_0 in $P_{\circlearrowleft}$.

Repeat until every $D \in \mathcal{D}$ is convex in Step a below.

a) If every $D \in \mathcal{D}$ is convex then Chop-Wedges(P) and exit.

b) If $p_i = \emptyset$ then let $(p_i, r, w) \leftarrow (b, a, c)$, where b is a reflex vertex of some dissection polygon D from $\mathcal{D}$, and a and c are the neighbors of b in D.

c) Let $\overrightarrow{s}$ be the ray emanating from p_i in direction of $\overrightarrow{wp_i}$ and let D be the dissection polygon where p_i is reflex.

d) If $\overrightarrow{s}$ hits a segment $\overline{qt} \subset D^\circ$ at a point $x \in \overline{qt}^{\ominus}$ and[2] t is interior to the triangle $\triangle(p_i, r, x)$ then rotate $\overrightarrow{s}$ towards q as described in Proposition 3.28.

e) If $\overrightarrow{s}$ hits a segment $\overline{qt} \subset D^\circ$ then

- $(P, u, \mathcal{D}) \leftarrow$ Extend-Reflex$(P, u, \mathcal{D}, p_i, r, \overrightarrow{s})$.
- If t is an anti-cap in $P_{\circlearrowleft}$ then let $(p_i, r, w) \leftarrow (t, a, q)$, where a is the other $(\neq q)$ neighbor of t in $P_{\circlearrowleft}$.
- Else let $(p_i, r, w) \leftarrow (q, a, t)$, where a is the other $(\neq t)$ neighbor of q in $P_{\circlearrowleft}$.

f) Else $\overrightarrow{s}$ hits an edge $\overline{p_k p_\ell}$ of D.

- If $\overline{p_k p_\ell}$ is good for D, p_i is a cap that is not part of an unsafe reflex twin, and not both $\overline{p_i r}$ and $\overline{p_i w}$ are good for D then $(P', u, \mathcal{D}) \leftarrow$ Drag-Edge$(P, u, \mathcal{D}, p_i, \overrightarrow{s}, \overline{p_k p_\ell})$.
  Else dissect D along $\overrightarrow{s}$ and let $P' \leftarrow P$.
- If $\measuredangle_D p_\ell$ is reflex[3] then let $(p_i, r, w) \leftarrow (p_\ell, a, c)$, where a is the other $(\neq p_k)$ neighbor of p_ℓ in $P_{\circlearrowleft}$ and c is the other $(\neq a)$ neighbor of p_ℓ in the dissection polygon of p_ℓ.
  Else let $p_i \leftarrow \emptyset$.
- Let $P \leftarrow P'$.

g) Saturate(P, u) and dissect $\mathcal{D}$ canonically.

Output: $(P, \mathcal{D})$.

[2] Assume without loss of generality that t and r are on the same side of $\overrightarrow{s}$.

[3] Assume without loss of generality that if $\measuredangle_D p_k$ is reflex then also $\measuredangle_D p_\ell$ is reflex. (Actually, p_k is always convex in D.)

An example illustrating the different steps of Algorithm 3.37 is provided in Figure 41 below.

As a first step towards proving the correctness of Algorithm 3.37 let us argue that it is indeed an algorithm.

Proposition 3.38 *Algorithm 3.37 terminates.*

Proof. Observe first that no basic operation ever removes a vertex from $V(P)$. If P is changed during Saturate in Step g then $|V(P)|$ increases by at least one because one endpoint of an unsaturated segment is added to $V(P)$. (More new vertices could appear along the geodesics.) Hence such changes can occur in a finite number of iterations only.

Otherwise, either Step e or Step f is executed in every iteration. In Step e $|V(P)|$ increases by at least two because the endpoints of a segment that was in P° are included as vertices into P. In Step f at least one reflex vertex of a region D from $\mathcal{D}$ is rendered convex and no new reflex vertices of a region from $\mathcal{D}$ are created. Note that new reflex vertices — in comparison to the previous iteration — may appear in the following call to Saturate in Step g, but as discussed above this can happen a finite number of times only. Hence, after a finite number of iterations every $D \in \mathcal{D}$ is convex and the algorithm terminates. □

As a second step, we show that shooting the ray from a "problematic" vertex makes this vertex safe. As we know from Proposition 3.24 that safe vertices remain safe, this means indeed that anti-caps and unsafe reflex twins can be resolved one by one.

Proposition 3.39 *Whenever a ray is shot from a reflex single vertex p_i of the frame in some iteration of Algorithm 3.37 such that p_i is an anti-cap or p_i is part of an unsafe reflex twin, the vertex p_i is safe at the end of that iteration.*

Proof. Note that Drag-Edge is not applied to a vertex that is an anti-cap or part of an unsafe reflex twin. Therefore either Extend-Reflex is applied to p_i and then p_i is a strictly convex vertex of the resulting frame and hence safe. Or we simply dissect along the ray shot from p_i and once again p_i is a safe vertex of the resulting frame. □

As for the first dissection algorithm described in Section 3.4.5, we show that the important properties of the polygon P and the dissection $\mathcal{D}$ are invariants of Algorithm 3.37.

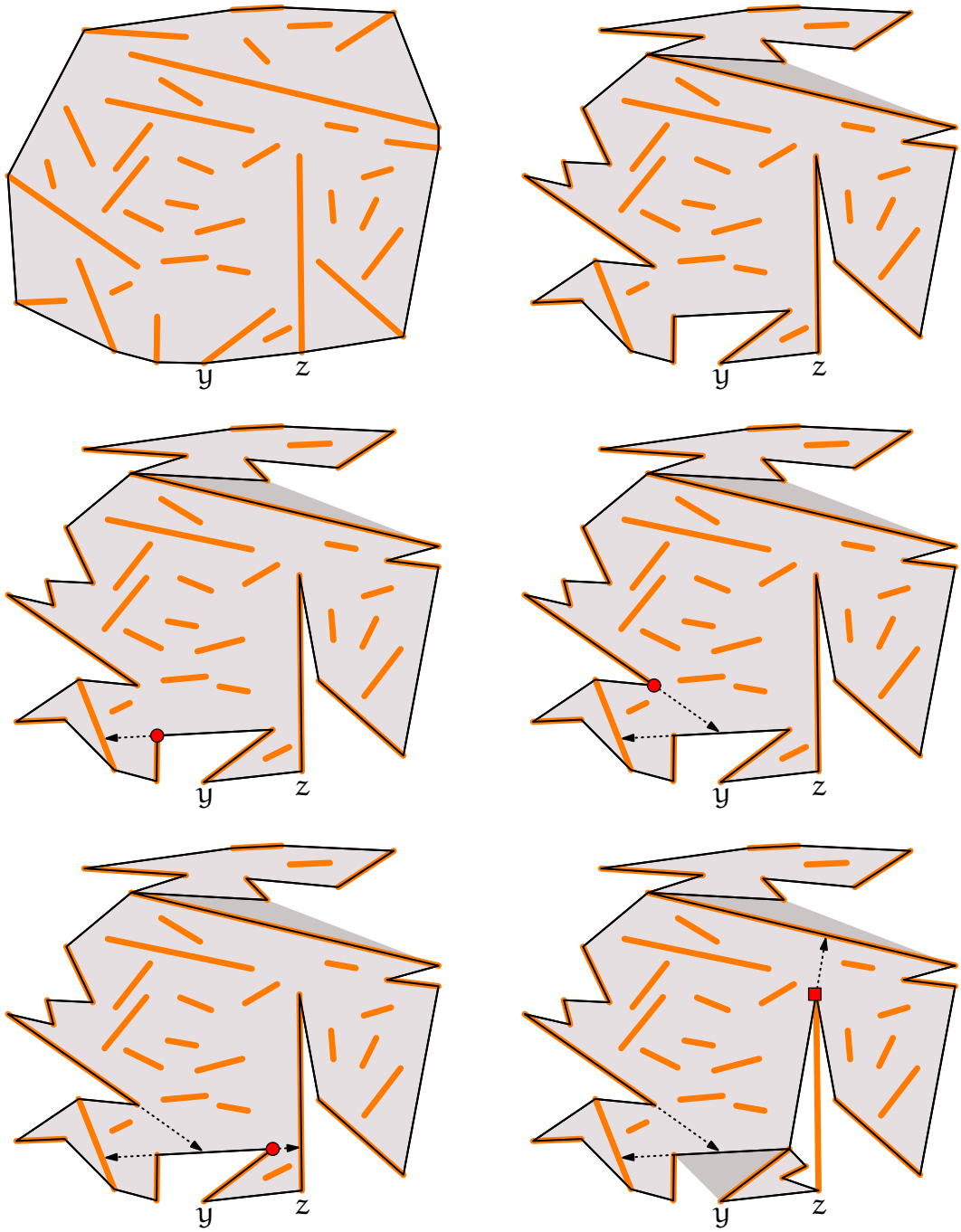


Figure 41: *Running Algorithm 3.37 on an example; wedges are shaded dark, and the points from which a ray is shot are marked: a circle denotes a cap, while a square stands for an anti-cap.*

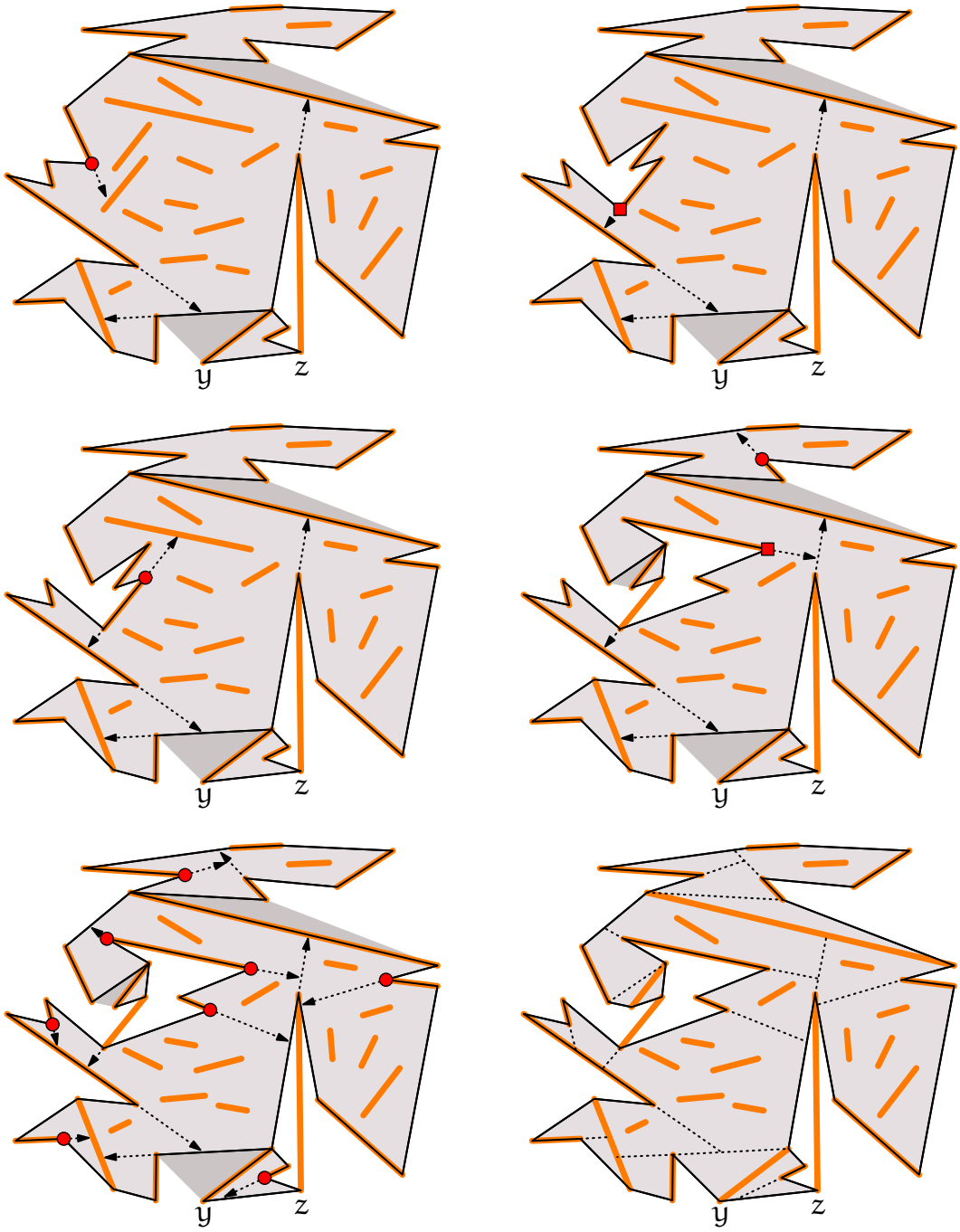


Figure 42: *Running Algorithm 3.37 on an example (continued). In the last step all wedges are chopped off and we obtain a dissection of* P *into convex polygons.*

Proposition 3.40 P *is always a frame and* $\mathcal{D}$ *is always a beautiful dissection of* P *before Step a in Algorithm 3.37. Moreover, the following invariants hold any time after Step b in Algorithm 3.37.*

(I1) If there exists an unsafe anti-cap in $P_{\circlearrowleft}$ *then it is* p_i*; if there exists an unsafe reflex twin in* $P_{\circlearrowleft}$ *then* p_i *is part of it;*

(I2) vertex r *is convex or safe in* $P_{\circlearrowleft}$*;*

(I3) for all vertices in $P_{\circlearrowleft}$ *that are reflex in some* $D \in \mathcal{D}$*, except for* p_i *and for the vertices of an unsafe reflex twin, both incident edges are perfect;*

(I4) if not both edges incident to p_i *in* $P_{\circlearrowleft}$ *are perfect then either* p_i *is part of an unsafe reflex twin in* $P_{\circlearrowleft}$ *or an edge incident to* p_i *was hit by the ray along which the dissection polygon was split in the previous iteration.*

Proof. P is always a frame and (*D1*) and (*D5*) hold for the same reasons as in Proposition 3.20. It remains to show that (*I1*)–(*I4*) hold any time after Step b and that (*D2**), (*D3**), and (*D4**) hold any time before Step a in Algorithm 3.37.

Let us argue that the statements hold for the first iteration of the loop in Algorithm 3.37.

(*D3**) is a direct consequence of Proposition 3.36. By Proposition 3.14 no double vertex of the frame is reflex in any dissection polygon. Thus every reflex vertex of any polygon $D \in \mathcal{D}$ is a single vertex of the frame and (*D4**) follows from Proposition 3.36. In fact, according to Proposition 3.36 there is at most one edge incident to a single reflex vertex of $P_{\circlearrowleft}$ that is not perfect: the edge between the two vertices that form a reflex twin, if existent. This implies (*I3*).

(*I1*), (*I2*), and (*I4*) are a consequence of the vertex selection after Careful-Saturate together with Proposition 3.36.

(*D2**) clearly holds before the first ray is shot: Before the call to Careful-Saturate every dissection polygon has at least three edges and surely one of them is different from $\overline{yz}$. If then a geodesic revisits a vertex, all resulting dissection polygons have a common edge with P along the geodesic and this edge is neither a wedge-edge (recall that if vertices are revisited then their original occurrence is labeled a wedge)

nor equal to $\overline{yz}$ because — as argued in Proposition 3.34 — y and z remain convex single vertices of the frame throughout Algorithm 3.37.

Thus indeed all claimed invariants hold for the first iteration of Algorithm 3.37. Next we will show that an arbitrary iteration maintains all invariants, provided that they hold in the iteration before.

In every iteration of Algorithm 3.37, exactly one of the following three events occurs: either Extend-Reflex is applied, or Drag-Edge is applied, or a dissection polygon is split along a ray. We will analyze these three cases separately and argue that the invariants are maintained for each of them.

But first let us make a general remark regarding a potential unsafe anti-cap or an unsafe reflex twin that exists in the frame at begin of the iteration.

According to (*D3**) there is at most one such configuration and by (*I1*) p_i is (part of) it. As the ray $\overrightarrow{s}$ in this iteration is shot from p_i, we conclude by Proposition 3.39 that any unsafe anti-cap or unsafe reflex twin that existed at begin of the iteration is safe at the end of the iteration.

Case 1: Suppose that Extend-Reflex is applied in an iteration of Algorithm 3.37.

(*D2**) holds at the end of this iteration because all resulting dissection polygons have a good edge along the constructed geodesics (in both Extend-Reflex and the following call to Saturate).

(*D3**) holds directly after Extend-Reflex in Step e as a consequence of (*I2*) and Proposition 3.26. Consider the following call to Saturate in Step g: As the orientation of the frame does not contain any alternation and the vertices along the geodesics are oriented towards the convex or safe vertices p_i and r, Saturate does not create any unsafe anti-cap or any unsafe reflex twin by Proposition 3.25. Thus (*D3**) holds at the end of this iteration.

Moreover, Proposition 3.26 tells us that if Extend-Reflex created an anti-cap then it is at one of q or t. The selection of p_i in Step e ensures that p_i is set to this anti-cap, if it exists. As p_i is set to either q or t, it is part of the only unsafe reflex twin: (q, t). Therefore (*I1*) and (*I4*) hold as well.

Also observe that if p_i is set to t in Step e then r is set to a vertex a on $\mathrm{geo}(t, x, p_i, r)$ that is convex or safe, even (or, rather, in particular)

if it is the same r as in the previous iteration. Similarly, if p_i is set to q then r is set to a vertex a on $\mathrm{geo}(p_i, x, q)$ that is convex, even if a is the same as the vertex p_i from the previous iteration. In both cases r is convex or safe in $P_{\circlearrowleft}$, that is, (*I2*) holds at the end of this iteration.

Regarding (*D4**) and (*I3*) suppose that a perfect edge $\overline{ac}$ incident to a reflex vertex a of some dissection polygon $A \in \mathcal{D}$ is not perfect anymore after Extend-Reflex. This can happen only if either a or c is revisited by a geodesic. If a is revisited then it is not a reflex vertex in any dissection polygon anymore by Proposition 3.14. On the other hand, it is impossible to revisit c and not also revisit a by a geodesic, unless c is also reflex in D. But then a and c form a reflex twin. We know from (*D3**) and (*I1*) that one of a or c is the vertex p_i to which we apply Extend-Reflex and p_i is a strictly convex single vertex of the frame after Extend-Reflex. Therefore, all edges incident to reflex vertices of D remain perfect, as long as the vertices remain reflex in D.

We may argue similarly for the following call to Saturate in Step g, noting that at that point the only unsafe reflex twin in the frame is (q, t). As they are covered by the exception in (*I3*), (*I3*) holds at the end of the iteration.

In order to prove (*D4**) we have to show that both q and t are incident to a perfect edge in the frame at the end of the iteration. Denote this frame by Q. Recall that by Proposition 3.29 q is a single reflex vertex of Q. Moreover, the other ($\neq t$) neighbor q_0 of q in $Q_{\circlearrowleft}$ is on the geodesic towards p_i (possibly, $q_0 = p_i$) and oriented away from q (if unsaturated). Therefore q_0 is a convex vertex in $Q_{\circlearrowleft}$ and as it is "guarded" by the strictly convex vertex p_i, the edge $\overline{q_0 q}$ is perfect. If t is a double vertex of Q then it is not reflex in any dissection polygon by Proposition 3.14 and hence there is nothing more to show. Otherwise, t is a single vertex of Q and we can argue as above that the edge $\overline{tt_0}$ is perfect, where t_0 is the other ($\neq q$) neighbor of t in $Q_{\circlearrowleft}$. (This time the vertices on the geodesic are oriented away from t and "guarded" by the strictly convex or safe vertex r in $Q_{\circlearrowleft}$.) Therefore, (*D4**) holds at the end of the iteration.

Case 2: Suppose that Drag-Edge is applied in an iteration of Algorithm 3.37.

(*D2**) holds at the end of this iteration because the second reflex occurrence of p_i created in Drag-Edge is safe. Therefore all resulting dissection polygons have a good edge along the constructed geodesics

(in both Drag-Edge and the following call to Saturate).

Regarding (*D3**) note that by (*D2**) and (*I1*) the vertices p_k and p_ℓ in Step f do not form an unsafe reflex twin. Thus — without loss of generality — p_ℓ is not part of an unsafe reflex twin and p_k is convex or safe in $P_\circlearrowleft$. Then the conditions from Proposition 3.32 are fulfilled and hence Drag-Edge does not create any unsafe reflex twin and at most one anti-cap, namely at p_ℓ. As the vertices along the geodesics constructed in Drag-Edge are oriented away from p_k and p_ℓ and the orientation does not contain any alternation, we conclude by Proposition 3.25 that the following call to Saturate in Step g does not create any unsafe reflex twin nor any unsafe anti-cap. This proves (*D3**).

If p_ℓ is a reflex vertex of some dissection polygon $D \in \mathcal{D}$ then p_ℓ was also reflex in the frame before Drag-Edge. As argued above, p_ℓ was not part of an unsafe reflex twin before Drag-Edge. Therefore its other ($\neq p_k$) neighbor a — that did not change in the course of Drag-Edge — is either convex or safe before and after Drag-Edge. This already proves (*I2*) in case that r is set during Step f.

Else p_ℓ is flat in D and hence it was reflex in D and by Proposition 3.14 also in $P_\circlearrowleft$ before Drag-Edge. As above, p_ℓ was not part of an unsafe reflex twin before Drag-Edge. Assuming without loss of generality that $p_\ell = p_{k\oplus 1}$ it follows that there is a subpath $(p_\ell, a = a_1, \ldots, a_m)$ in $P_\circlearrowleft$, for some $m \in \mathbb{N}$, where a_i is flat, for all $1 \leq i < m$, and a_m is safe. On the other side, p_ℓ is "guarded" by the second and safe reflex occurrence of p_i that was created in Drag-Edge. Thus p_ℓ is not part of any anti-cap if it is flat in D.

In summary we have shown that if Drag-Edge created an unsafe anti-cap then p_i is set to this anti-cap in Step f. This proves (*I1*).

On the other hand, if p_ℓ is not reflex in D after Drag-Edge then p_i is set in Step b at begin of the next iteration. As there is no unsafe reflex twin in the frame, any neighbor of the chosen reflex vertex p_i is either convex or safe. This completes the proof for (*I2*).

It remains to consider (*I3*), (*I4*), and (*D4**). For the vertices that are revisited by some geodesic we may argue as above in Case 1. The argument is even slightly simpler because this time there is no unsafe reflex twin in the frame at all. The original occurrence of p_i is labeled as a wedge, but it does not appear as a reflex vertex of any dissection polygon anymore. If p_ℓ is still reflex in D then p_i is set to p_ℓ in Step f and both edges incident to p_ℓ are perfect: for $\overline{p_\ell a}$ we have already

argued above that it did not change during this iteration, and for the other edge along the geodesic towards the safe occurrence of p_i this is ensured by orienting the neighbor of p_ℓ away from p_ℓ. Again this neighbor of p_ℓ is convex or safe (if it is p_i) in the frame and "guarded" by the safe occurrence of p_i. This proves (*I3*), (*I4*), and (*D4**).

Case 3: Suppose that a dissection polygon $D \in \mathcal{D}$ is simply split along a ray $\overrightarrow{s}$ shot from some reflex vertex p_i of D in an iteration of Algorithm 3.37.

Let us first consider (*D2**).

If both edges incident to p_i are good for D then clearly they are good edges for both resulting dissection polygons.

On the other hand, by (*D4**) at least one of the edges incident to p_i is good for D. Moreover, (*I4*) tells us that if not both edges incident to p_i are good for D then either one of the edges was hit by the ray in the previous iteration or p_i is part of an unsafe reflex twin in $P_{\circlearrowleft}$.

Let us first consider the case of reflex twins. Recall that both vertices of the reflex twin (potentially) created during the initial call to Careful-Saturate are single vertices of the frame for which all incident edges are good by Proposition 3.36. The only other operation that creates an unsafe reflex twin in $P_{\circlearrowleft}$ is Extend-Reflex, namely at (q, t). We have argued in Case 1 above that the other ($\neq \overline{qt}$) edge $\overline{qq_0}$ incident to q is perfect. Similarly, we have shown that if t is a single vertex of P then the other ($\neq \overline{qt}$) edge $\overline{tt_0}$ incident to t is perfect. Clearly in this case also $\overline{qt}$ is good. It remains to consider the case where t appeared twice on the geodesic in Extend-Reflex or t is revisited by a geodesic during the following call to Saturate. Although $\overline{qt}$ is a wedge-edge in this case, there is good edge $\overline{tt_1}$ for D incident to t along the geodesic that revisited t. (Recall that the vertices of D along this geodesic are oriented away from the reflex occurrence of t in $P_{\circlearrowleft}$ and "guarded" by the safe vertex r with possibly $r = t_1$. Therefore t_1 is convex or safe in the resulting frame.) As in this case the next ray $\overrightarrow{s}$ is shot from q in direction $\overrightarrow{tq}$, this ray cannot hit $\overline{tt_1}$. (Not even after a potential rotation of $\overrightarrow{s}$ in Step (d): after such a rotation either Extend-Reflex is applied or an edge incident to a reflex vertex of D is hit; but both t and t_1 are convex or safe.) Thus, there is a good edge on both sides of $\overrightarrow{s}$: on one side $\overline{qq_0}$ and on the other side $\overline{tt_1}$.

It remains to consider the case that an edge incident to p_i was hit by a ray $\overrightarrow{e}$ shot from some reflex vertex a in the immediately preceding

iteration. This means that no operation was applied in the previous iteration and hence p_i is a cap that is not part of an unsafe reflex twin. Also, $\overrightarrow{s}$ did not hit a good edge because then Drag-Edge would have been applied. On the other hand, by (*I3*) (for the previous iteration) the edge incident to p_i that was hit by $\overrightarrow{e}$ was certainly good (before it was hit). There are two possible implications:

Either both edges incident to a are good: As the ray $\overrightarrow{s}$ from p_i is shot in direction of the edge that was hit by $\overrightarrow{e}$, it cannot hit back to $\overrightarrow{e}$, not even after a possible rotation of $\overrightarrow{s}$ in Step (d). (Recall that after such a rotation either Extend-Reflex is applied or an edge incident to a reflex vertex of D is hit.) The good edge incident to a and bounding D was not hit by $\overrightarrow{s}$, either. (As argued above that would have invoked Drag-Edge.) Hence, there is a good edge on both sides of $\overrightarrow{s}$: on one side the edge incident to p_i that was not hit by $\overrightarrow{e}$ and on the other side the good edge incident to a.

Or a is an anti-cap or part of an unsafe reflex twin. If both edges incident to a are good then we can argue as above. Else a was generated by Extend-Reflex, that is, a is the segment endpoint q of the segment $\overline{qt}$ to which we extended in the previous iteration and t is a double vertex of the resulting frame. Let q_0 denote the other ($\neq t$) neighbor of $a = q$ and denote the two dissection polygons that are generated by the split along $\overrightarrow{e}$ by D_1 and D_2 where $q_0 \in V(D_1)$ and $t \in V(D_2)$. We have already argued above that in this case $\overline{qq_0}$ is good for D_1 and there is always an edge $\overline{tt_1}$ incident to t that is good for D_2. We have to show that after one of the D_1 or D_2 is split along $\overrightarrow{s}$ there is still a good edge in both resulting dissection polygons.

Suppose that D_1 is split, that is, $p_i \in V(D_1)$. As above, $\overrightarrow{s}$ cannot hit back to $\overrightarrow{e}$ and it does not hit $\overline{qq_0}$, either, because $\overline{qq_0}$ is a good edge and then Drag-Edge would have been applied. Hence, there is a good edge on both sides of $\overrightarrow{s}$: on one side the edge incident to p_i that was not hit by $\overrightarrow{e}$ and on the other side the good edge incident to $q = a$.

Otherwise, D_2 is split and $p_i \in V(D_2)$. Again $\overrightarrow{s}$ does not hit $\overrightarrow{e}$ nor $\overline{tt_1}$, but we also have to show that $\overrightarrow{s}$ does not hit the wedge-edge $\overline{qt}$, either. As both q and t are strictly convex vertices of D_2, no rotation of $\overrightarrow{s}$ in Step (d) will make $\overrightarrow{s}$ hit $\overline{qt}$. Moreover, $\overrightarrow{e}$ is shot in direction of $\overrightarrow{tq}$ and $\overrightarrow{s}$ in direction of the edge hit

by $\overrightarrow{e}$. Hence, if $\overrightarrow{e}$ is not rotated in Step (d) then the claim is obvious as well. It remains to consider a possible rotation of $\overrightarrow{e}$. Note that such a rotation rotates $\overrightarrow{e}$ towards t (and away from q_0). As the rotation stops at p_i, p_i is a vertex of D_1, contrary to our assumption $p_i \in V(D_2)$. Therefore, $\overrightarrow{s}$ does not hit $\overline{qt}$ and there are good edges on both sides of $\overrightarrow{s}$: on one side the edge incident to p_i that was not hit by $\overrightarrow{e}$ and on the other side the good edge $\overline{tt_1}$.

Thus (*D2**) holds throughout Algorithm 3.37.

(*D3**) holds clearly because there is no unsafe reflex twin and no unsafe anti-cap at the end of this iteration. Thus, also (*I1*) holds trivially and similarly (*I2*) holds if p_i and r are set in Step b.

Regarding (*D4**) and (*I3*) observe that p_i is safe after this iteration and whenever an edge incident to a reflex vertex of D is hit then p_i is set to this reflex vertex. By (*D3**) and (*I1*) no ray can hit an edge that is incident to an unsafe reflex twin. This proves (*D4**), (*I3*), and (*I4*). It also implies (*I2*) in case that p_i and r are set in Step f.

This concludes the case analysis and proves that the claimed invariants hold all through Algorithm 3.37. □

It remains to analyze the behaviour of Chop-Wedges in the last step of Algorithm 3.37. By our labeling scheme every double vertex of the frame is labeled as a wedge exactly once. We have to show that these vertices are really wedges as defined in Definition 3.21.

Corollary 3.41 *During Algorithm 3.37 an anti-cap is never labeled as a wedge.*

Proof. An anti-cap created during the initial call to Saturate is a single vertex of the resulting frame by Proposition 3.36 that is safe after the first iteration as argued above in Proposition 3.40. According to Proposition 3.24 safe vertices remain safe throughout Algorithm 3.37. (We proved Proposition 3.24 for Algorithm 3.18. But it is easily seen to hold for Algorithm 3.37 as well because the basic operations applied in both algorithms are the same.)

An unsafe anti-cap created by Extend-Reflex is not revisited during the following call to Saturate by Corollary 3.31. The vertices along the geodesic in Extend-Reflex are oriented towards the strictly convex

vertex p_i (along $\mathrm{geo}(p_i, x, q)$) or towards the safe (due to (*I2*)) vertex r. As the orientation of the frame does not contain any alternation at that point, we may conclude by Proposition 3.25 that the following call to Saturate does not create any unsafe anti-cap.

Finally, consider an unsafe anti-cap created by Drag-Edge. According to Proposition 3.32 it can appear at p_ℓ only. By Corollary 3.33 the vertex p_ℓ is not revisited during the following call to Saturate. As the next ray is shot from p_ℓ if p_ℓ is reflex in D, p_ℓ becomes safe in the next iteration. We have argued in the proof of Proposition 3.40 that p_ℓ is not part of any anti-cap if it is flat in D. Therefore, an unsafe anti-cap created by Drag-Edge remains a single vertex of the frame throughout Algorithm 3.37. □

3.8 Induction

This section completes the inductive proof of Theorem 3.2. The proof is rather straightforward, as most work has already been done during the development of Algorithm 3.37. Let us summarize the properties of the frame P and the dissection $\mathcal{D}$ computed by Algorithm 3.37.

Corollary 3.42 *For a set S of disjoint line segments, not all collinear, and an edge $\{y, z\}$ of $\mathrm{conv}(\partial S)$, there is a simple frame P for S and a collection $\mathcal{D}$ of pairwise non-overlapping convex polygons such that the following conditions are satisfied.*

- *(C1) $\{y, z\}$ is an edge of P;*
- *(C2) for every $s \in S$, either $s \subset P^\circ$ and there is a $D \in \mathcal{D}$ such that $s \subset D^\circ$, or $V(s) \subset V(P)$ and $s \cap D^\circ = \emptyset$, for all $D \in \mathcal{D}$;*
- *(C3) $\bigcup_{D \in \mathcal{D}} \mathcal{R}(D) \subseteq \mathcal{R}(P)$;*
- *(C4) every polygon $D \in \mathcal{D}$ has a common edge with P that is different from $\{y, z\}$.*

Proof. Denote by $(P, \mathcal{D})$ the output of Algorithm 3.37 for input S and $\{y, z\}$. By Proposition 3.40 P is a frame before Chop-Wedges is applied in Step a and by Proposition 3.22 the result is a simply connected polygon satisfying (*F1*)–(*F3*).

At the end of the preceding iteration of Algorithm 3.37 Saturate has been called in Step g. (Similarly, before the first iteration Careful-Saturate has been called at the very beginning.) As Chop-Wedges does not change $V(P)$, all segments from S are saturated in P and hence (*F5*) is fulfilled.

According to our wedge labeling scheme every double vertex in $P_{\circlearrowleft}$ is labeled exactly once as a wedge. By Corollary 3.41 all vertices labeled as a wedge correspond to caps, that is, they are wedges as defined in Definition 3.21. Therefore, all vertices labeled as a wedge are removed in Chop-Wedges and P is a simple polygon. As there exists no double vertex in P, (*F4*) is trivially fulfilled as well and thus P is indeed a frame.

Observe that $\mathcal{D}$ is not necessarily a dissection of P because $\mathcal{R}(P)$ increases if Chop-Wedges chops off any wedge. But clearly all polygons in $\mathcal{D}$ are convex by the end condition of Algorithm 3.37 and they are contained in $\mathcal{R}(P)$ because they formed a dissection of the frame before Chop-Wedges was applied.

$\{y, z\}$ is an edge of P due to Proposition 3.34. (*C2*) is an easy consequence of (*D5*), (*F1*) and (*F2*). Finally, (*C4*) is implied by (*D2**). □

Using these properties we can prove Theorem 3.2 inductively as follows.

Proof. [**of Theorem 3.2**] We prove by induction on $|S|$ the following statement. For a set S of disjoint line segments, not all collinear, and for any fixed edge $\{y, z\}$ of $\mathrm{conv}(\partial S)$, there is a Hamiltonian polygon H for S such that $\{y, z\}$ is an edge of H.

The statement holds for $|S| = 2$. Suppose it holds for all S' with $1 < |S'| < |S|$.

Consider the simple frame P and the dissection $\mathcal{D}$ described in Corollary 3.42. If both endpoints of every segment are in $V(P)$ then the statement holds. If there is a segment s whose neither endpoint is in $V(P)$ then by Property (*C2*) s is in the interior of some $D \in \mathcal{D}$. By property (*C4*) D has a common edge $\overline{ab} \neq \overline{yz}$ with P. By (*C2*) and because D is convex we have $C(D) := \mathrm{conv}(\partial S \cap D^\circ) \subset D^\circ$. Moreover, $C(D)$ has an edge $\overline{cd}$ such that both $\overline{ac}$ and $\overline{bd}$ are non-crossing visibility edges. If $\overline{c_1 d_1}, \ldots, \overline{c_m d_m}$, for some $m \in \mathbb{N}$, are the segments in D° and they are all collinear in this order, then replace the edge $\overline{ab}$ of P

by the path $(a, c_1, d_1, \ldots, c_m, d_m, b)$. Otherwise, there is by induction a Hamiltonian polygon $H(D)$ for the segments from S which lie in D° such that $\overline{cd}$ is an edge of $H(D)$. Let $H^*(D)$ be the path from c to d that traverses $H(D)$ except for the edge $\overline{cd}$. Replace the edge $\overline{ab}$ of P by the path $(a, c) \cdot H^*(D) \cdot (d, b)$. Doing so for each $D \in \mathcal{D}$ that contains segments from S results in a Hamiltonian polygon for S. □

Up to now we have been working under the assumption that all segments in S are non-degenerate. But the generalization to possibly degenerate line segments is rather straightforward. We can prevent any ray that is shot during Algorithm 3.37 from hitting a degenerate segment by perturbing it infinitesimally. Then a degenerate segment is either visited by a geodesic or it appears on some convex hull boundary. In both cases the segment is immediately saturated; in particular, no degenerate segment ever appears as a reflex vertex of the frame. This proves that Theorem 3.2 holds as stated, for any set of disjoint line segments that are not all collinear.

3.9 Runtime Analysis

In this section we show that Theorem 3.2 provides a polynomial time algorithm to construct Hamiltonian polygons for disjoint line segments.

Theorem 3.43 *For any set of n pairwise disjoint line segments, not all collinear, a Hamiltonian polygon can be constructed in $O(n^2)$ time and $O(n)$ space.*

Proof. Represent the frame and its dissection collectively in a *doubly connected edge list* (DCEL) [65]. In this way an edge can be replaced by a polygonal path in time proportional to the size of the path. As each vertex of this planar subdivision is incident to at most four edges, we can examine the neighborhood of vertices and edges in constant time.

Store for each input segment the incident vertices of the frame (if any), and, vice versa, for each vertex of the frame store its (unique) incident input segment. For each dissection polygon store the list of input segments that are interior to the polygon.

During Algorithm 3.37 we maintain a list of those vertices that are unsaturated, a list of those vertices that are reflex in some dissection

polygon, and a list of wedge-edges. Clearly, the overall space needed by these data structures is linear.

The convex hull computations can be done in $O(n^2)$ time altogether using Jarvis' Wrap [55]. (Any vertex can appear on at most one of the convex hull boundaries.) The bridge edges used to connect the inductively constructed partial solutions to the global frame can be obtained brute-force in overall linear time.

The geodesics can be computed in a brute-force manner spending linear time per vertex. As each vertex can appear on at most two geodesics during Algorithm 3.37, the overall time needed is again $O(n^2)$. Similarly, the ray shooting and a possible rotation of the ray are done brute-force in linear time per ray and, hence, $O(n^2)$ time overall. (Rays are shot from reflex vertices of dissection polygons only. After such a ray has been processed the corresponding vertex is safe and does not appear as a reflex vertex of a dissection polygon anymore.)

The vertices that appear along the constructed geodesics have to be marked as (un–)saturated and have to be inserted or removed from the corresponding list. Similarly, vertices are marked if they are reflex in some dissection polygon, and edges are marked as wedge-edges if an incident vertex is labeled as a wedge. Finally, at the end of Algorithm 3.37 all wedges are removed. As all these events can occur only once per vertex or edge, they can be processed in overall linear time.

It remains to consider the time spent to maintain the lists of segments interior to the dissection polygons. Every time a dissection polygon is split, we have to test for each of the interior segments in which of the two resulting polygons it is contained. This decision can be made in linear time by computing the next polygon edge below one of the segment endpoints. As the number of splits is linear as well, the overall time bound for these tests is $O(n^2)$. □

Several of the tasks discussed above can be implemented in subquadratic time using more complicated data structures. The main bottleneck seems to be the computation of geodesics. The construction of many geodesics has been a recent focus of research [32, 12]. But in our algorithm the geodesics to be constructed in an iteration often depend on the result of the previous iteration. Therefore, it is not clear how to obtain a sufficiently large set of geodesics for bulk-processing.

3.10 Remarks

As mentioned in Chapter 1, it is well known that there are triangulations that do not contain a Hamiltonian cycle [26]. As we may consider the edges of a triangulation as a set of intersecting segments, this also shows that one cannot simply drop the disjointness condition from Theorem 3.2. In fact, Figure 43 shows a configuration of seven segments for which not even the visibility graph is Hamiltonian: three segments form a triangle that contains the fourth segment, and the remaining three segments are placed along the convex hull such that no triangle edge is an edge of the convex hull, but the vertices of the triangle remain convex hull vertices.

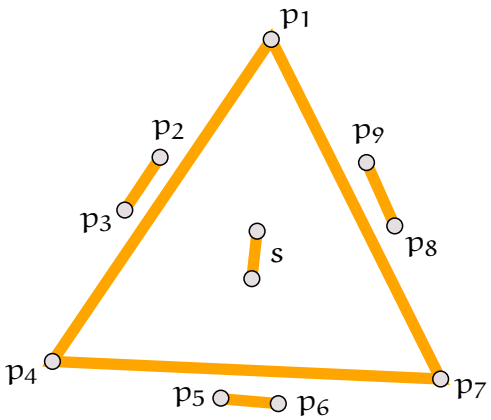


Figure 43: *A set of segments for which the visibility graph is not Hamiltonian.*

Theorem 3.44 *There is a set S of seven pairwise non-overlapping line segments in the plane for which $\mathrm{Vis}(S)$ is not Hamiltonian.*

Proof. Denote the vertices along the convex hull by $p_1, \ldots, p_9$, as indicated in Figure 43. Removal of p_1 and p_4 disconnects $\mathrm{Vis}(S)$ into two components one of which is $\{p_2, p_3\}$. That is, any Hamiltonian cycle in $\mathrm{Vis}(S)$ must visit p_2 and p_3 in between p_1 and p_4. Analogously, p_8 and p_9 must be visited in between p_7 and p_1, and p_5 and p_6 must be visited in between p_4 and p_7. But then there is no way to visit the vertices of the interior segment s. Therefore, there is no Hamiltonian cycle in $\mathrm{Vis}(S)$. □

Another interesting problem is the generalization of Theorem 3.2 to polygons. If we consider line segments as (convex) polygons on two

vertices, the next interesting class of polygons are triangles. Recall that a Hamiltonian polygon for a set of triangles may not enter the interior of any triangle. The question about the existence of Hamiltonian polygons for disjoint triangles is still open.

Conjecture 3.45 *For any finite set of pairwise disjoint triangles, not all collinear, there exists a Hamiltonian polygon.*

The statement is not true in general for quadrilaterals, as the example in Figure 44 shows. The example consists of three large squares s_1, s_3, and s_5, three medium squares s_2, s_4, and s_6, and seven small squares $A, \ldots, G$. In Figure 44, the small squares are drawn much larger than they are ideally in order to increase their visibility. Observe that for the large squares three vertices are on the convex hull, whereas for the medium squares and for A, B, and C two vertices are on the convex hull boundary. At the center, the three large squares are very close together such that, for example, s_6 and F cannot see the center vertex c_1 of s_1, and the center square G cannot see anything except for the three center vertices c_1, c_3, and c_5 of the large squares.

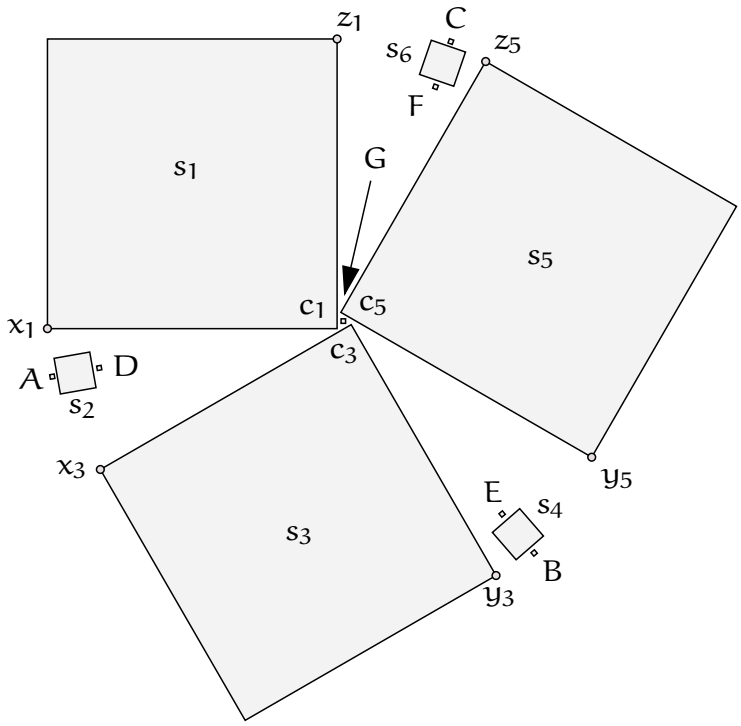


Figure 44: *A set of disjoint squares for which the visibility graph is not Hamiltonian.*

Theorem 3.46 *There is a set of thirteen pairwise disjoint squares in the plane such that the visibility graph of their vertices with respect to the squares is not Hamiltonian.*

Proof. Let H be a Hamiltonian cycle in the visibility graph $\mathrm{Vis}(Q)$ for the set Q of squares shown in Figure 44. Consider the three convex hull vertices of the large square s_1. One of these vertices has degree two in $\mathrm{Vis}(Q)$ and, hence, it is appears in between the other two on H. Analogously, the convex hull vertices of s_3 and s_5, respectively, are visited directly after each other. Similarly, the vertices of the small squares A, B, and C must be visited in between the two convex hull vertices of s_2, s_4, and s_6, respectively.

The vertices in $X := V(A) \cup V(D) \cup V(s_2)$ are connected to the remaining vertices in three ways only: via the convex hull vertex x_1 of s_1, via the convex hull vertex x_3 of s_3, and via the center vertex c_1. Observe that it is impossible to visit all vertices of X on a path between x_1 and x_3 without visiting c_1. Thus, c_1 is connected to at least one vertex of X in H. Furthermore, x_1 must be connected to either a vertex from X or c_1 in H, and if x_1 is connected to a vertex from X in H then x_3 must be connected to one of c_1 or c_3 in H. Altogether, in H there are at least two edges between the three center vertices c_1, c_3, and c_5 and the vertices from $X \cup \{x_1, x_3\}$.

A similar argument for the set $Y := V(B) \cup V(s_4) \cup V(E)$ shows that in H there are at least two edges between the three center vertices c_1, c_3, and c_5 and the vertices from $Y \cup \{y_3, y_5\}$. Analogously, there are at least two edges between the three center vertices c_1, c_3, and c_5 and the vertices from $V(C) \cup V(s_6) \cup V(F) \cup \{z_1, z_5\}$. But then there is no way to include the vertices of the center square G into H, in contradiction to H being a Hamiltonian cycle.

Therefore, there is no Hamiltonian cycle in $\mathrm{Vis}(Q)$. □

Chapter 4

Alternating Paths

This chapter is concerned with *alternating paths* through disjoint line segments in the plane.

Definition 4.1 *Consider a set S of n non-degenerate disjoint line segments in the plane. A simple path $P = (v_1, \ldots, v_k)$, $k \in \mathbb{N}$, in $\mathrm{Vis}(S)$ is called an* **alternating path** *if and only if it consists of segment edges and visibility edges in alternating order, that is, $\overline{v_{2i-1}v_{2i}} \in S$, for every $1 \leq i \leq \lfloor k/2 \rfloor$, or $\overline{v_{2i}v_{2i+1}} \in S$, for every $1 \leq i \leq \lfloor (k-1)/2 \rfloor$.*

Note that in particular any path of size at most two in $\mathrm{Vis}(S)$ is alternating. Figure 45 shows an example for an alternating path through some disjoint line segments.

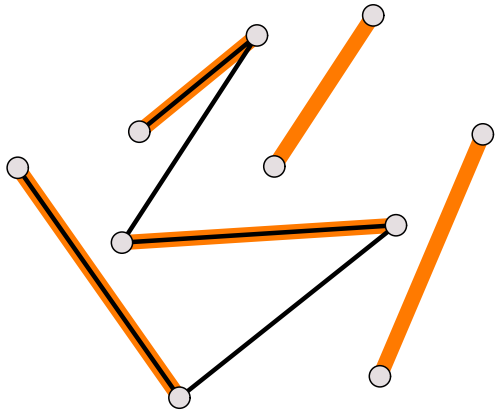


Figure 45: *An alternating path through disjoint segments.*

In particular, we are interested in the following question: What is the maximum number of vertices that can be visited by an alternating path? The following theorem states that there is always an alternating path of size logarithmic in the number of segments.

Theorem 4.2 *For any set S of n disjoint line segments in the plane there is an alternating path in $\mathrm{Vis}(S)$ that visits $2\lceil \log_2(n+2) \rceil - 3$ vertices.*

Apart from a constant factor, this is best possible:

Theorem 4.3 *For any $n_0 \in \mathbb{N}$, there exists an $n > n_0$ and a set S of n disjoint line segments in the plane, such that $\mathrm{Vis}(S)$ does not contain an alternating path visiting more than $\frac{12}{\log_2 3} \log_2 n - 17 < 7.57 \log_2 n$ vertices.*

The proof of Theorem 4.2 uses the existence of a Hamiltonian polygon for S that was proven in Chapter 3. Moreover it is algorithmic and can hence be used to construct an alternating path of size at least $2\log_2(n+2) - 3$. However, note that the constructed path is not necessarily the largest alternating path for the given set of line segments. The problem of computing the largest alternating path for a given set of segments may be (much) more difficult.

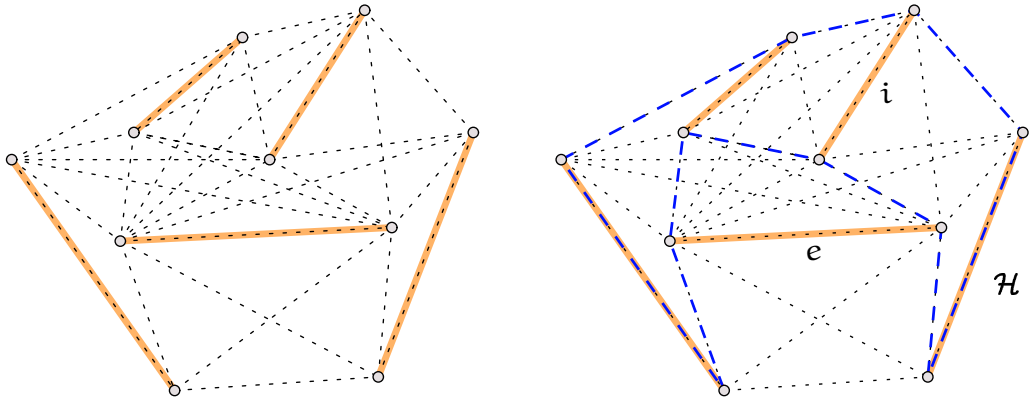
The proof of Theorem 4.3 simply specifies a family of sets of disjoint segments whose visibility graph does not contain an alternating path on more than $7.57 \log_2 n$ vertices.

4.1 Lower Bound

As already mentioned, the bound stated in Theorem 4.2 relies on the fact that every segment endpoint visibility graph contains a Hamiltonian polygon, that is, Theorem 3.2. We consider a planar subgraph of the segment endpoint visibility graph, which is a union of a Hamiltonian cycle and a complete matching (corresponding to the segments of S) on the segment endpoints. As this graph is planar, all paths in it are simple.

Recall that Theorem 3.2 states that $\mathrm{Vis}(S)$ contains a Hamiltonian polygon $\mathcal{H}$ if not all segments in S are collinear. Clearly, if all seg-

ments are collinear then there is an alternating path through all segment endpoints. Otherwise, we can rely on the existence of a Hamiltonian polygon $\mathcal{H}$. Observe that $\mathcal{H}$ is not necessarily alternating because it may contain several visibility edges in a row (see Figure 46(b) for an example). $\mathcal{H}$ can possibly consist of visibility edges only.



(a) $\mathrm{Vis}(S)$. (b) $\mathcal{H}$.

Figure 46: *A segment endpoint visibility graph and one of its Hamiltonian polygons.*

A segment $s \in S$ which is not in $\mathcal{H}$ is necessarily a diagonal or epigonal of $\mathcal{H}$. In Figure 46(b), for example, segment i is a segment diagonal, while segment e is a segment epigonal of $\mathcal{H}$. Denote by $\mathcal{D}$ the Hamiltonian polygon $\mathcal{H}$ together with all its segment diagonals and epigonals, and denote $V := V(\mathcal{D})$ and $E := E(\mathcal{D})$ in the following.

Proposition 4.4 *Every vertex of $\mathcal{D}$ has degree two or three. If $\deg(v) = 2$ for a vertex v, then no segment diagonal or epigonal is incident to $\mathcal{H}$ at v, therefore v is incident to a visibility edge and a segment edge. If $\deg(v) = 3$ then v is incident to a segment diagonal or epigonal and to two visibility edges along $\mathcal{H}$.* □

Observe that $\mathcal{D}$ is a PSLG. Hence any path in $\mathcal{D}$ is a simple path in $\mathrm{Vis}(S)$.

We show in the next lemma, that one can build an alternating path from any segment edge to any vertex in $\mathcal{D}$.

Lemma 4.5 *For every segment $\overline{e_0 e_1} \in S \subset E$ and every vertex $f \in V$ the graph $\mathcal{D} = (V, E)$ contains an alternating path $(e_0, e_1, \ldots, e_k = f)$, for some $k \in \mathbb{N}$.*

The remainder of this section is devoted to the proof of Lemma 4.5. The proof of Theorem 4.2 then follows by elementary arguments.

Define a distance function d on the vertex set V as follows. For any $v \in V$, $v \neq e_0$, let $d(v)$ be the size of the smallest (not necessarily alternating) path connecting v and f along $\mathcal{H}$ that does not pass through e_0. (Such a path always exists, since $\mathcal{H}$ is a cycle.) If $e_0 = f$, let $d(e_0) := 0$, else $d(e_0) := \infty$. Next, we orient all visibility edges in $\mathcal{D}$ such that they are directed towards the vertex with smaller value $d(\cdot)$. Two examples are depicted in Figure 47. Note that we do not consider $\mathcal{D}$ as a directed graph, the orientation induced by $d(\cdot)$ is merely an aid to construct paths.

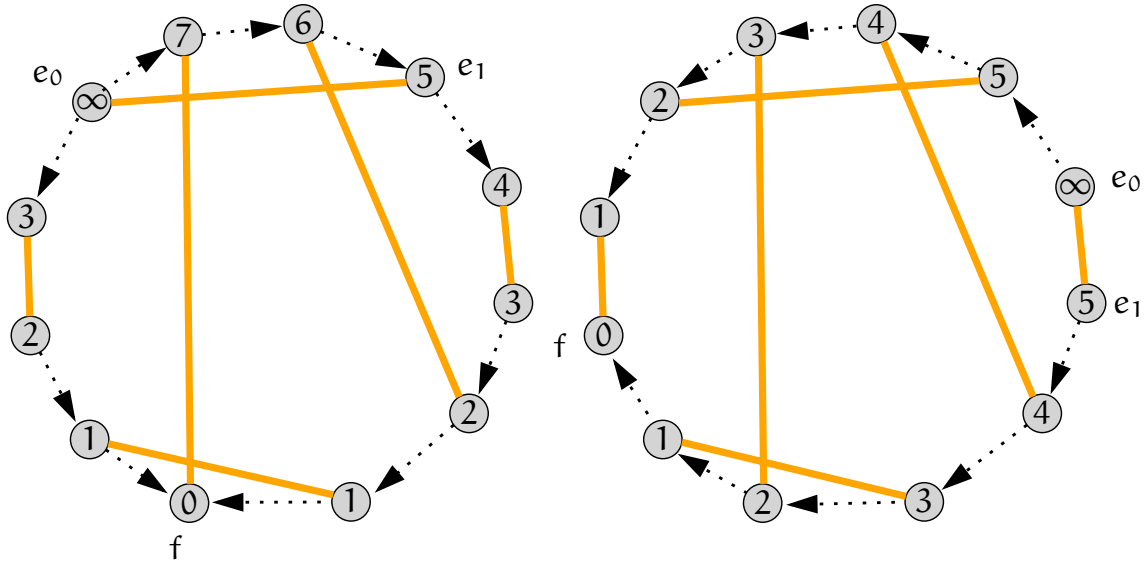


Figure 47: *Two examples with orientations and distances. In this drawing, we emphasize the Hamiltonian structure of $\mathcal{D}$; there are no crossings in the original* PSLG.

Proposition 4.6 *With respect to the orientation according to $d(\cdot)$, every vertex of degree three in $\mathcal{D}$ is incident to at least one outgoing and at most one incoming visibility edge, except for f which might be incident to two incoming visibility edges.* □

With help of this orientation, we can try to build an alternating path P in $\mathcal{D}$ starting from (e_0, e_1) and directed towards f as follows.

Algorithm 4.7

Input: an edge (v_0, v_1) of $\mathcal{D}$ and a set $X \subseteq V$ of vertices.

Initialization: $P \leftarrow (v_0, v_1)$.

Algorithm:
While P is a path and has not reached any vertex from X do

a) $(u, v) \leftarrow$ last edge of P.

b) If $\deg_{\mathcal{D}}(v) = 2$ then append the other ($\neq u$) neighbor of v in $\mathcal{D}$ to P.

c) Else if $\{u, v\}$ is a visibility edge then append the unique segment edge incident to v in $\mathcal{D}$ to P.

d) Else append a visibility edge outgoing (according to $d(\cdot)$) from v to P.

Output: P.

To check that Algorithm 4.7 is well defined, refer to Propositions 4.4 and 4.6. When called with (e_0, e_1) and $\{f\}$, the algorithm either terminates by reaching f, or by the path P reaching a vertex for the second time and thus becoming a walk. (The algorithm terminates, because the number of edges in P strictly increases in every iteration.) If the path does not reach f, we are left with a path connected to a cycle, which looks like a balloon with a cord attached to it. Let us derive a more formal — and slightly more general — description for this type of configuration.

Definition 4.8 *A subgraph* G *of* $\mathrm{Vis}(S)$ *is called* **walkable** *from a vertex* $v \in V(G)$ *if and only if for every vertex* $u \in V(G) \setminus \{v\}$*, there is an alternating path within* G *from* v *to* u *whose edge incident to* u *is a segment edge.*

Note that, in particular, a graph consisting of a single vertex or two vertices connected by a segment edge always form a walkable subgraph.

Definition 4.9 *The union* $B = G \cup P$ *of two subgraphs of* $\mathrm{Vis}(S)$ *with* $V(G) \cap V(P) = \{v\}$ *and* $E(G) \cap E(P) = \emptyset$ *is called a* **balloon** *if and only if* G *is walkable from* v*, and* $P = (v = v_0, v_1, \ldots, v_k = u)$*,* $k \in \mathbb{N}$*, is an alternating path in* $\mathrm{Vis}(S)$ *such that* $\overline{vv_1} \in S$*. We call* $\mathrm{src}(B) := u$ *the* **source**, $\mathrm{hrt}(B) := v$ *the* **heart**, $\mathrm{bdy}(B) := G$ *the* **body**, *and* $\mathrm{cor}(B) := P$ *the* **cord** *of* B.

Figure 48 shows an example of a balloon.

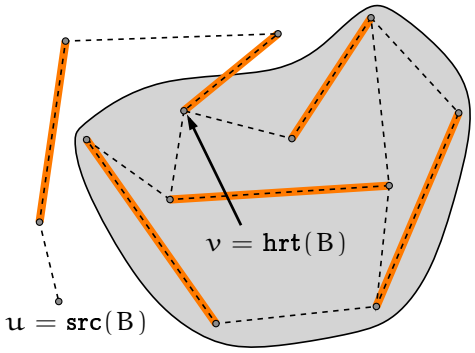


Figure 48: *A balloon* B*; the body (shaded) is walkable from* v*.*

Proposition 4.10 *Consider the path* P *computed by Algorithm 4.7 for the input* (v_0, v_1) *and* X *where* $f \in X$*. Then either* P *reaches a vertex from* X*, or* P *forms a cycle, or* P *forms a balloon.*

Proof. If the algorithm does not reach any vertex from X, it terminates with a walk

$$P = (v_0, v_1, \ldots, v_s, v_{s+1}, \ldots, v_k = v_s),$$

for some $0 \leq s < k$, such that $(v_0, \ldots, v_{k-1})$ is a path. If $s = 0$, then P forms a cycle. Otherwise, we claim that both $\overline{v_{k-1}v_k}$ and $\overline{v_s v_{s+1}}$ are visibility edges.

Indeed, a segment edge is included into the path directly after one of its incident vertices has been reached. Hence $\overline{v_{k-1}v_k}$ cannot be a segment edge, otherwise the algorithm would have selected v_{k-1} instead of v_{s+1} as a successor of v_s. If $\overline{v_s v_{s+1}} \in S$ then the preceding edge $\overline{v_{s-1}v_s}$ is a visibility edge with $d(v_s) < d(v_{s-1})$ by construction. For the same reason, the edge $\overline{v_{k-1}v_k}$ is directed towards $v_k = v_s$. Then Proposition 4.6 tells us that $v_s = f$, and the algorithm would have stopped there.

Proposition 4.4 implies that every second edge in P is a segment edge; in particular, if $\overline{v_s v_{s+1}}$ is a visibility edge then $\overline{v_{s-1}v_s}$ is a segment edge. Thus, every vertex on the path $(v_{s+1}, \ldots, v_{k-1})$ can be reached from v_s on an alternating path that ends with a segment edge: either via v_{s+1} or via v_{k-1} along P. Altogether, we have shown that the constructed path P forms a balloon with source v_0 and heart v_s. □

Proposition 4.11 *For any vertex* v *in the body of a balloon* B*, there*

is an alternating path in B *from* $\mathrm{hrt}(B)$ *to* v *which starts with a visibility edge and ends with a segment edge.*

Proof. Since the segment edges are pairwise disjoint, there can only be one segment edge incident to any vertex. The segment edge incident to $\mathrm{hrt}(B)$ is part of $\mathrm{cor}(B)$ by definition; hence, there are only visibility edges incident to $\mathrm{hrt}(B)$ in $\mathrm{bdy}(B)$. The claim follows from the fact that $\mathrm{bdy}(B)$ is walkable. □

Definition 4.12 *A sequence* $\mathcal{B} = (B_1, B_2, \ldots, B_\ell)$, $\ell \in \mathbb{N}_0$, *is called a* **balloon-path** *in* $\mathcal{D}$ *if and only if it satisfies the following conditions.*

1. *For any* i, $1 \leq i \leq \ell$, B_i *is a balloon in* $\mathrm{Vis}(S)$.
2. *For any* i, $1 \leq i < \ell$ *and any* j, $i < j \leq \ell$,

$$V(B_i) \cap V(B_j) = \begin{cases} \mathrm{src}(B_j) \in V(\mathrm{bdy}(B_i)) & \text{, if } j = i+1 \\ \emptyset & \text{, otherwise,} \end{cases}$$
$$E(B_i) \cap E(B_j) = \emptyset\,.$$

Denote $|\mathcal{B}| := \ell$, $V(\mathcal{B}) := \bigcup_{i=1}^{\ell} V(B_i)$, $\mathrm{src}(\mathcal{B}) := \mathrm{src}(B_1)$, *and* $\mathrm{bdy}(\mathcal{B}) := \bigcup_{i=1}^{\ell} \mathrm{bdy}(B_i)$.

We observe a few immediate consequences of this definition. Consider a balloon-path $\mathcal{B} = (B_1, B_2, \ldots, B_\ell)$ in $\mathcal{D}$.

Proposition 4.13 *For any* i, $2 \leq i \leq \ell$, *the edge incident to* $\mathrm{src}(B_i)$ *in* B_i *is a visibility edge.*

Proof. Since $\mathrm{src}(B_i) \in V(\mathrm{bdy}(B_{i-1}))$ by definition, there is an alternating path from $\mathrm{src}(B_{i-1})$ to $\mathrm{src}(B_i)$ in B_{i-1} that ends with a segment edge. There is exactly one segment edge incident to every vertex in $\mathrm{Vis}(S)$, and $E(B_{i-1}) \cap E(B_i) = \emptyset$. Thus, any edge incident to $\mathrm{src}(B_i)$ in B_i must be a visibility edge. □

Note that Proposition 4.13 implies that $|V(\mathrm{cor}(B_i))| \geq 3$ for any $2 \leq i \leq \ell$.

Proposition 4.14

(i) For every vertex $u \in V(\mathcal{B})$*,* $\mathcal{B}$ *contains an alternating path from* $\mathrm{src}(B_1)$ *to* u*.*

(ii) For every vertex $u \in \mathrm{bdy}(\mathcal{B})$*,* $\mathcal{B}$ *contains an alternating path from* $\mathrm{src}(B_1)$ *to* u *that ends with a segment edge.*

Proof. The statement is obvious for $u \in V(\mathrm{cor}(B_1))$. Otherwise, there is by definition an alternating path in B_1 from $\mathrm{src}(B_1)$ to $\mathrm{hrt}(B_1)$ that ends with a segment edge. By Proposition 4.11, any vertex in $\mathrm{bdy}(B_1)$ can be reached from $\mathrm{hrt}(B_1)$ within B_1 on an alternating path starting with a visibility edge and ending with a segment edge. Since both paths can be concatenated to a single alternating path, we are done for the case that $u \in V(\mathrm{bdy}(B_1))$. Otherwise, we can use the same argument for $\mathrm{src}(B_2)$, that lies in $\mathrm{bdy}(B_1)$ by definition, to construct an alternating path from u to $\mathrm{src}(B_2)$ which ends with a segment edge. Then $(B_2, \ldots, B_\ell)$ forms again a balloon-path, which is strictly smaller than $\mathcal{B}$ by Proposition 4.13. The claim follows by induction on $|\mathcal{B}|$. □

Proposition 4.15 *For any vertex* $v \in V(\mathcal{B}) \setminus \{\mathrm{src}(B_1)\}$ *the segment edge* $\overline{uv}$ *incident to* v *is an edge of* $\mathcal{B}$*.*

Proof. If $v \in V(\mathrm{bdy}(\mathcal{B}))$, the claim follows from Proposition 4.14 (ii). So, let $v \in V(\mathrm{cor}(B_i))$ for some $1 \leq i \leq \ell$: if $v = \mathrm{src}(B_i)$ and $i > 1$, we have $v \in V(\mathrm{bdy}(B_{i-1}))$; if $v = \mathrm{hrt}(B_i)$, it is $v \in V(\mathrm{bdy}(B_i))$. In the remaining case, $\deg_{\mathrm{cor}(B_i)}(v) = 2$. Since $\mathrm{cor}(B_i)$ is an alternating path, one of the edges incident to v in $\mathrm{cor}(B_i)$ must be a segment edge. □

We have now collected all tools to describe an algorithm to construct a balloon-path from (e_0, e_1) headed towards f, that will provide the proof of Lemma 4.5.

Algorithm 4.16

Input: an edge (e_0, e_1) of $\mathcal{D}$ and a vertex $f \in V$.

Initialization:

$\mathcal{B}$	$\leftarrow$	$()$.	*(balloon path)*
(κ, μ)	$\leftarrow$	(e_0, e_1).	*(start edge)*

Algorithm:

Repeat the following until $f \in V(\mathcal{B})$ in Step e below.

a) Let $P := (v_1 = \kappa, v_2 = \mu, \ldots, v_k)$ be the output of Algorithm 4.7 applied to (κ, μ) and $\{f\} \cup V(\mathcal{B})$.

b) $(B_1, \ldots, B_\ell) \leftarrow \mathcal{B}$.

c) If $v_k \in V(B_i)$, for some $1 \leq i \leq \ell$, then

$$\mathcal{B} \leftarrow \left(B_1, \ldots, B_{i-1}, P \cup \bigcup_{j=i}^{\ell} B_j \right) .$$

d) Else $\mathcal{B} \leftarrow (B_1, \ldots, B_\ell, P)$.

e) If $f \in V(\mathcal{B})$ then exit.

f) $\kappa \leftarrow$ a vertex from $\mathrm{bdy}(B_{|\mathcal{B}|})$ that minimizes $d(\cdot)$.

g) $\mu \leftarrow$ a vertex with $\{\kappa, \mu\} \in E(\mathcal{D})$ and $d(\mu) < d(\kappa)$. (cf. Proposition 4.6)

Output: $\mathcal{B}$.

Note that the choice of κ in Step f is not necessarily unique because two nodes might have the same $d(\cdot)$ value; in that case we can break the tie arbitrarily.

Proposition 4.17 *At the beginning of any iteration of the loop in Algorithm 4.16, $\mathcal{B}$ forms a balloon-path with source e_0.*

Proof. The statement is trivial for the first iteration, since at that point $\mathcal{B} = ()$. In the second iteration, Algorithm 4.7 is called with parameters (e_0, e_1) and $\{f\}$. According to Proposition 4.10, if the path P does not reach f then it forms either a cycle or a balloon. The function $d(\cdot)$ is defined such that no visibility edge is directed towards e_0, unless $e_0 = f$. The segment edge $\overline{e_0 e_1}$ incident to e_0 is already in P from begin on; hence it is not possible to revisit e_0 along a segment edge, either. Therefore, the path P returned by Algorithm 4.7 in the second iteration cannot be a cycle: it must form a balloon.

Assume $\mathcal{B} = (B_1, \ldots, B_\ell)$ is a balloon-path at begin of some iteration. Algorithm 4.7 returns a path $P = (v_1 = \kappa, v_2 = \mu, \ldots, v_k)$ which,

according to Proposition 4.10, either reaches a vertex from $\{f\} \cup V(\mathcal{B})$, or forms a balloon. Notice that if P forms a cycle, it necessarily reaches a vertex of $V(\mathcal{B})$, since $\kappa = v_k \in V(\mathrm{bdy}(\mathcal{B}))$.

Let us first consider the case that P reaches a vertex $v \in V(\mathcal{B})$. We claim that $\overline{v_{k-1}v_k}$ is a visibility edge: If $v \neq e_0$, recall that by Proposition 4.15 the segment edge $\overline{vw}$ incident to v lies in $\mathcal{B}$ as well. Thus, Algorithm 4.7 stops if P reaches w. Similarly, the segment edge $\overline{e_0e_1}$ incident to e_0 is part of $\mathcal{B}$, and the algorithm stops if P reaches e_1. Now there are two subcases to consider.

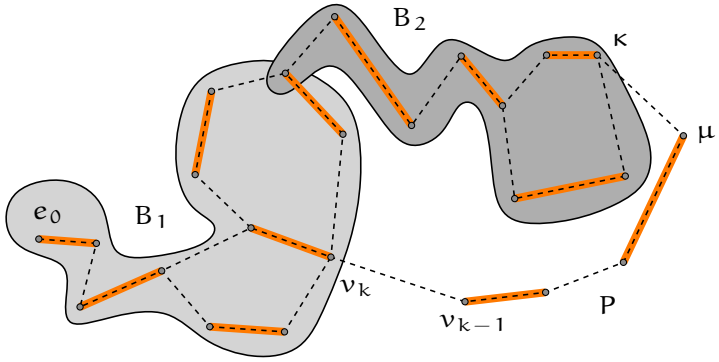
(1) $v_k \in V(\mathrm{bdy}(B_i))$ (Figure 49(a)), for some $1 \leq i \leq \ell$. There is, by definition, an alternating path from $\mathrm{src}(B_i)$ to v_k that ends with a segment edge. Hence, any vertex $v_2, \ldots, v_{k-1}$ can be reached from $\mathrm{src}(B_i)$ on an alternating path ending with a segment edge: either via v_k and P or via the balloon-path $(B_i, \ldots, B_\ell)$ to κ and then P (see Proposition 4.14 (ii)). (Note that if P forms a cycle then $v_k = \kappa$, and the argument still goes through. In fact, one could show that P never reaches κ again in the course of Algorithm 4.16.) Furthermore, the same argument can be applied to the vertices in $\bigcup_{j=i+1}^{\ell} V(\mathrm{cor}(B_j))$. Thus, $P \cup \bigcup_{j=i}^{\ell} B_j$ is a balloon with source $\mathrm{src}(B_i)$ and heart $\mathrm{hrt}(B_i)$.

(2) $v_k \in V(\mathrm{cor}(B_i))$ (Figure 49(b)), for some $1 \leq i \leq \ell$. Let

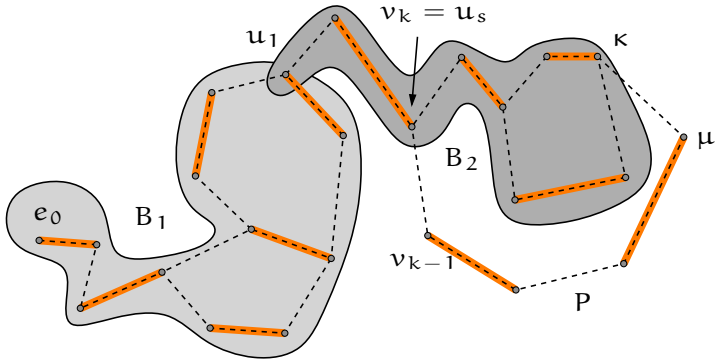
$$\mathrm{cor}(B_i) = (u_1 = \mathrm{src}(B_i), \ldots, u_s = v_k, \ldots, u_r) ,$$

for some $1 < s \leq r$. Note that for $s = 1$ we have $u_s = \mathrm{src}(B_i) \in V(\mathrm{bdy}(B_{i-1}))$ and can thus argue as in Case 1. (Recall that P does not revisit e_0.) Since all paths in $\mathcal{B}$ are constructed using Algorithm 4.7, all visibility edges in the cords of the balloons in $\mathcal{B}$ are oriented from the source to the heart of the balloon. Hence, we can argue as in Proposition 4.10 that $\overline{u_{s-1}u_s} \in S$. By the same reasoning as above, $P \cup \bigcup_{j=i}^{\ell} B_j$ is a balloon with source $\mathrm{src}(B_i)$ and heart $u_s = v_k$.

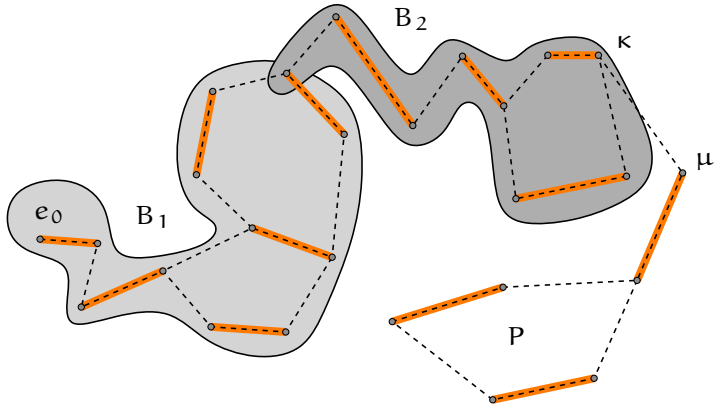
It remains to consider the else-branch, that is, the case that the path P that is constructed recursively by Algorithm 4.7 hits itself before reaching any vertex from $V(\mathcal{B})$ (Figure 49(c)). Then by Proposition 4.10 P either reaches f or it forms a balloon with $\mathrm{src}(P) = \kappa$. If P reaches f, the algorithm terminates; otherwise, $(B_1, \ldots, B_\ell, P)$ forms again a balloon-path because $\kappa \in \mathrm{bdy}(B_\ell)$. □



(a) P hits the body of a balloon.



(b) P hits the cord of a balloon.



(c) P hits itself.

Figure 49: *Illustrations for Algorithm 4.16.*

Algorithm 4.16 immediately provides the proof for Lemma 4.5.

Proof. [of Lemma 4.5] We apply Algorithm 4.16 to (e_0, e_1) and f. If we can show that the algorithm always terminates, the claim follows from Propositions 4.17 and 4.14 (i). Strictly speaking, after termination $\mathcal{B} = (B_1, \ldots, B_\ell)$ is not always a balloon-path anymore; but $(B_1, \ldots, B_{\ell-1})$ is a balloon-path and B_ℓ is an alternating path whose one endpoint lies in $\mathrm{bdy}(B_{\ell-1})$ and is incident to a visibility edge in B_ℓ. Hence, the argument from Proposition 4.14 goes through.

To show termination, note first that no edge is ever discarded from $\mathcal{B}$, that is, $|E(\mathcal{B})|$ increases monotonely over the execution of the algorithm. Moreover, this increase is strict because in every iteration at least the edge $\{\kappa, \mu\}$ is added to $E(\mathcal{B})$. □

It might be worthwhile to note that we did not use anywhere the fact that $\mathcal{D}$ is planar. The proof of Theorem 4.2 is completed by the following elementary argument.

Proof. [of Theorem 4.2] Consider an arbitrary directed segment $\overrightarrow{e_0e_1} \in S$. For $i \in \mathbb{N}$ denote by V_i the set of vertices in V that can be reached on an alternating path on exactly $i+1$ vertices, starting with (e_0, e_1). For example, $V_1 = \{e_1\}$. According to Proposition 4.4, we have $|V_i| \leq 2|V_{i-1}|$ for i even, and $|V_i| \leq |V_{i-1}|$ for i odd. Setting $n_i := |V_i|$ yields thus

$$n_i \leq \begin{cases} 2^{\frac{i-1}{2}} & , i \text{ even}, \\ 2^{\frac{i}{2}} & , i \text{ odd}. \end{cases}$$

Hence, $\sum_{i=1}^{k} |V_i| \leq 2^{\frac{k+3}{2}} - 3$ for k odd, and $\sum_{i=1}^{k} |V_i| \leq 3 \cdot 2^{\frac{k}{2}} - 3$ for k even.

By Lemma 4.5, there exists an ℓ, $1 \leq \ell \leq n$, such that $V \setminus \{e_0\} = \bigcup_{i=1}^{\ell} V_i$. Combining this with the inequality from above yields $n-1 = |V \setminus \{e_0\}| \leq \sum_{i=1}^{\ell} |V_i| \leq 3 \cdot 2^{\frac{\ell}{2}} - 3$, that is, $\ell \geq 2\log_2(n+2) - 2\log_2 3$. The claim follows, since the graph contains an alternating path on at least $\ell + 1$ vertices. □

The above proof suggests a simple algorithm to construct a long alternating path.

Corollary 4.18 *For any set S of n disjoint line segments in the plane and any Hamiltonian polygon H for S an alternating path in $\mathrm{Vis}(S)$*

that visits at least $2\lceil\log_2(n+2)\rceil - 3$ *vertices can be computed in* $O(n)$ *time.*

Proof. Construct the graph $\mathcal{D}$ that consists of H together with all segment diagonals and epigonals. Start an alternating breadth first search from an arbitrary segment edge (e_0, e_1). (If the distance of the current vertex in the bfs-tree to e_0 is odd, consider segment edges only, otherwise consider visibility edges only.) As argued above, the last vertex visited during this search is reached on an alternating path on at least $2\lceil\log_2(n+2)\rceil - 3$ vertices in the bfs-tree. As a planar graph $\mathcal{D}$ contains a linear number of edges (in fact, there are at most $3n$ edges) and thus the search can be done in linear time. □

4.2 Upper bound

Complementing the results from the previous section, we show here an asymptotically matching upper bound, that is, we construct sets $\mathcal{S}_k$, $k \in \mathbb{N}$, of disjoint line segments that do not have large alternating paths. We remark that an $O(\log n)$ bound was already known by a construction due to Urrutia [87], but with a larger constant coefficient than ours.

Proof. [of Theorem 4.3] We construct the sets of segments $\mathcal{S}_k$, $k \in \mathbb{N}$, recursively as follows. All line segments are chords of a circle c. $\mathcal{S}_1$ consists of three segments arranged in a triangular fashion, i.e., such that $\mathrm{Vis}(\mathcal{S}_1) \cong K_6$. The endpoints of the chords partition c into arcs. $\mathcal{S}_k$ is obtained from $\mathcal{S}_{k-1}$ by inserting a sequence of three segments (i.e., a copy of $\mathcal{S}_1$) on every arc of c that is bounded by only one segment of $\mathcal{S}_{k-1}$. Figure 50 shows $\mathcal{S}_1$ and $\mathcal{S}_2$.

The two endpoints of any segment in this construction are adjacent to the same set of segment endpoints in the visibility graph $\mathrm{Vis}(\mathcal{S}_k)$. Hence, we can interpret $\mathrm{Vis}(\mathcal{S}_k)$ as complete ternary tree of depth $k-1$ where each vertex is formed by a clique of three segments (Figure 50(c)). Let λ_k be the size of a largest alternating path in $\mathcal{S}_k$. Since the largest simple path in a tree of depth $k-1$ has size $2k-1$ and since visiting a 3-clique of segments means visiting 6 vertices, we conclude that $\lambda_k =$

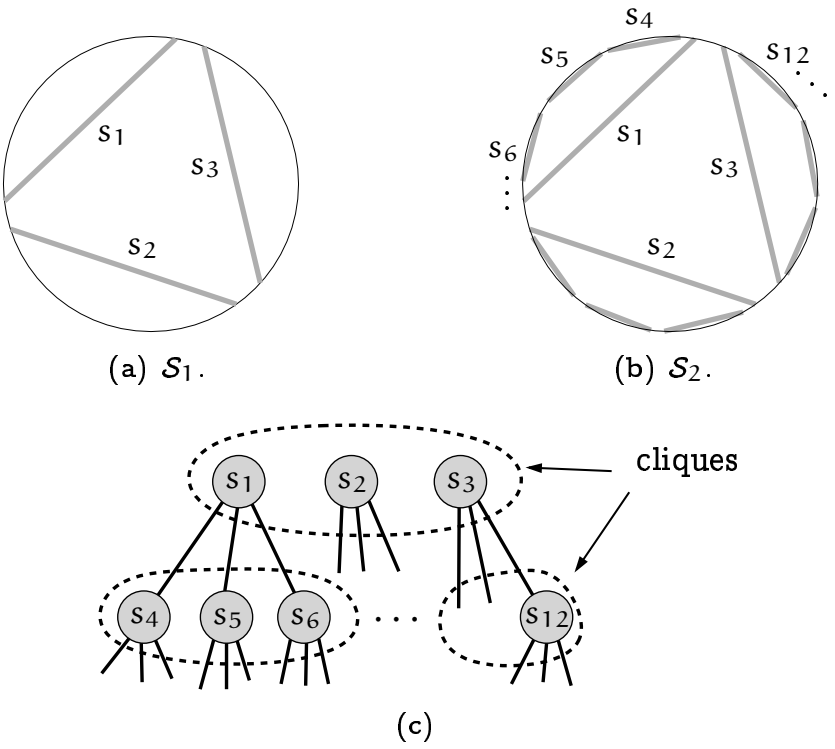


Figure 50: *The construction of* $\mathcal{S}_k$.

$12k-6$. $\mathcal{S}_k$ contains exactly $n_k := 3^{k+1}-3$ vertices. Hence,

$$\begin{aligned}\lambda_k &= 12\left(\log_3 n_k - 1 + \log_3 \frac{3^k}{3^k-1}\right) - 6\\ &= \frac{12}{\log_2 3}\log_2 n_k - 18 + 12\log_3 \frac{3^k}{3^k-1}\end{aligned}$$

and the claimed result follows because the last term is less than one for $k \geq 3$. □

Note that the choice to construct $\mathcal{S}_k$ using groups of three segments is not arbitrary: Suppose $\mathcal{S}_k$ is constructed using groups of x segments, for some positive real number x. Then the calculation from above yields

$$\lambda_k = 4x\left(\log_x n_k + \log_x \frac{x-1}{2x} + \log_x \frac{x^k}{x^k-1}\right) - 2x\,.$$

To minimize λ_k for sufficiently large k, we have to minimize $x/\ln x$. This term is minimized for $x = e \approx 2.718$ and three is the closest integer.

Chapter 5

Chordless Paths

This chapter discusses the parametric complexity of finding chordless paths in graphs. Recall the definition of the problem Cp3v.

Problem 5.1 (Cp3v) *Given an undirected graph* $G = (V, E)$*, a positive integer* k*, and three distinct vertices* $s, t, v \in V$*, is there a chordless* (s, v, t)*-path of size at most* k *in* G*?*

First we show in Section 5.1 that Cp3v is a member of the complexity class $W[1]$ using a reduction to Short Nondeterministic Turing Machine Computation. The complementing $W[1]$-hardness result, a reduction from Independent Set, is then subject of Section 5.2. Combining both reductions yields the main theorem of this chapter.

Theorem 5.2 Cp3v *is* $W[1]$*-complete with respect to* k*.*

Section 5.3 and Section 5.4 discuss extensions of these results to several related problems regarding the existence of chordless paths and cycles.

5.1 Membership in W[1]

In this section we analyze the parameterized complexity of Cp3v and prove the problem to be in $W[1]$ with respect to its natural parameter, the path's size. First note that a chordless (s, t)-path P is already

determined by its set of vertices. For example, only one neighbor x of s can be in $V(P)$ because any appearance of another neighbor in $V(P)$ would introduce a chord. Similarly, exactly one neighbor of x (other than s) can be in $V(P)$. In this manner P can be uniquely reconstructed from $V(P)$.

Proposition 5.3 *A subgraph P of G is a chordless (s,t)-path if and only if P is connected, s and t have degree one in $G[V(P)]$, and all vertices other than s and t have degree two in $G[V(P)]$.* □

We do not know how to directly reduce Cp3v to Weighted q-Cnf-Satisfiability. Instead, we reduce to a different problem called Short Nondeterministic Turing Machine Computation or SnTmc, for short, that is defined below. SnTmc is known to be $W[1]$-complete [17, 31], and reduction to SnTmc and its relatives has proven to be a useful tool to establish membership results within the W-hierarchy [19, 20].

Problem 5.4 (Short Nondeterministic Turing Machine Computation)
Given a single-tape (two-way infinite), single-head nondeterministic Turing machine M, a word x on the alphabet of M, and a positive integer k, is there a computation of M on input x that reaches a final accepting state in at most k steps?

Theorem 5.5 Cp3v *is in $W[1]$ with respect to k.*

Proof. Consider an instance (G, s, v, t, k) of Cp3v, where $G = (V, E)$ is an undirected graph, $s, v, t \in V$, and k is a positive integer. We will construct an instance (M, k') of SnTmc such that there is a computation for M that reaches a final accepting state in at most $k' = k^2 + 4k$ steps if and only if there exists a chordless (s, v, t)-path of size at most k in G. (It is important that k' depends on k only and not on n.) A schematic view of the construction is shown in Figure 51.

Let $M = (\Sigma, Q, \Delta, g_1, \{A\})$, where the alphabet Σ is defined as $\Sigma := \{\square\} \cup \{\sigma_u \mid u \in V\}$; the state set is

$$Q := \{A, R\} \cup \{g_i \mid 1 \le i \le k\} \cup \{a, b, c, d, l, r\} \cup \{p_u \mid u \in V\} \cup \{q_u \mid u \in V\};$$

the transition relation $\Delta : Q \times \Sigma \times Q \times \Sigma \times \{+, -, 0\}$ is defined below; the initial state is g_1; the final accepting state is A, and the final rejecting state is R. When the Turing machine starts, all tape cells contain the

blank symbol ($\square$). The computation consists of three phases: first, the at most k vertices of a chordless (s, v, t)-path P in G are "guessed" by writing the sequence of corresponding symbols σ_u, $u \in V(P)$, onto the tape. The next two phases are completely deterministic and check that

(ii) P visits s, v, and t in the order given, and

(iii) P is a chordless path in G.

In the following paragraphs we describe the three phases of the computation. For each step, we first give a verbal description and then define the corresponding formal transition of the Turing machine subsequently.

First Phase. The Turing machine may write up to k arbitrary vertex symbols onto the tape: $(g_i, \square, g_{i+1}, \sigma_u, +) \in \Delta$, for all $u \in V$ and all $1 \leq i < k$, and $(g_i, \square, a, \sigma_u, 0) \in \Delta$, for all $u \in V$ and all $1 \leq i \leq k$. (The transition specifies, in order, current state, symbol under the head, new state after transition, symbol to write to the tape, and movement of the head: $+$ for right, $-$ for left, and 0 for stay.) After the first phase, the Turing machine is in state a and the sequence of between one and k vertex symbols starts at the current tape cell, extending to the left.

Second Phase. Check whether the guessed sequence visits t, v, and s, in order. The rightmost symbol should be σ_t: $(a, \sigma_t, b, \sigma_t, -) \in \Delta$. Then somewhere σ_v: $(b, \sigma_u, b, \sigma_u, -) \in \Delta$, for all $u \in V \setminus \{v\}$, and $(b, \sigma_v, c, \sigma_v, -) \in \Delta$. The leftmost symbol is σ_s: $(c, \sigma_u, c, \sigma_u, -) \in \Delta$, for all $u \in V \setminus \{s\}$, $(c, \sigma_s, d, \sigma_s, -) \in \Delta$, and $(d, \square, l, \square, +) \in \Delta$. For all state/symbol combinations that are not explicitly mentioned (for example, $(b, \square)$ or (a, σ_v)) there is a transition to the final rejecting state R. After the second phase, the machine is in state l and the head points towards the leftmost of the symbols that have been guessed in Phase 1. The content of the tape remains unchanged during Phase 2.

Third Phase. Scan and remove the first vertex: $(l, \sigma_u, p_u, \square, +) \in \Delta$, for all $u \in V$. If no more vertices are left at this point, we are done: $(p_u, \square, A, \square, 0) \in \Delta$, for all $u \in V$. Else the next vertex has to be adjacent: $(p_u, \sigma_w, q_u, \sigma_w, +) \in \Delta$, for all $u, w \in V$ for which $\{u, w\} \in E$. Whatever follows must not be adjacent: $(q_u, \sigma_w, q_u, \sigma_w, +) \in \Delta$, for all $u, w \in V$ with $u \neq w$ and $\{u, w\} \notin E$. If all vertices have been

checked, return to the leftmost: $(q_u, \square, r, \square, -) \in \Delta$, for all $u \in V$, and $(r, \sigma_w, r, \sigma_w, -) \in \Delta$, for all $w \in V$. Finally, re-iterate: $(r, \square, l, \square, +) \in \Delta$. Again, all state/symbol combinations that are not explicitly mentioned lead to the final rejecting state R. Note that after the third phase, all tape cells contain the blank symbol again.

Correctness. Phase 3 ensures that all vertices guessed in Phase 1 are distinct, as, otherwise the right scan in state q_u, for some $u \in V$, fails. Moreover, because of the transition from p_u to q_u, the vertices chosen form a path P in G. The right scan in state q_u also ensures that no two of the vertices are connected except along P. Finally, in Phase 2 we check that the endpoints of P are s and t, and that P visits v. Altogether, the machine reaches an accepting state if and only if it guesses the vertices of a chordless (s, v, t)-path of size at most k in Phase 1. An easy calculation reveals that if k symbols are written onto the tape in Phase 1, the computation consists of exactly k^2+4k transitions. Clearly, M can be constructed in time polynomial (quadratic) in both n and k. □

5.2 Hardness for W[1]

In this section, we prove that Cp3v is $W[1]$-hard using a reduction from Independent Set, which is one of the "classical" $W[1]$-hard problems [29].

Problem 5.6 (Independent Set) *Given a positive integer* k *and an undirected graph* $G = (V, E)$*, is there an independent set of size at least* k *in* G*?*

Consider an instance of Independent Set, that is, a graph $G = (V, E)$ and an integer k, $1 \leq k \leq |V|$, and let $V = \{v_1, \ldots, v_n\}$. We construct a graph G' from G such that the answer to the Cp3v problem on G' provides the solution to the independent-set problem on G.

The main ingredient for our construction is called *vertex choice diamond*. It consists of n vertices $v_1^i, \ldots, v_n^i$ plus two extra vertices s^i and t^i connected to each of the n vertices v_j^i, $1 \leq j \leq n$, as shown in Figure 52. Clearly, there are exactly n chordless (s^i, t^i)-paths in such

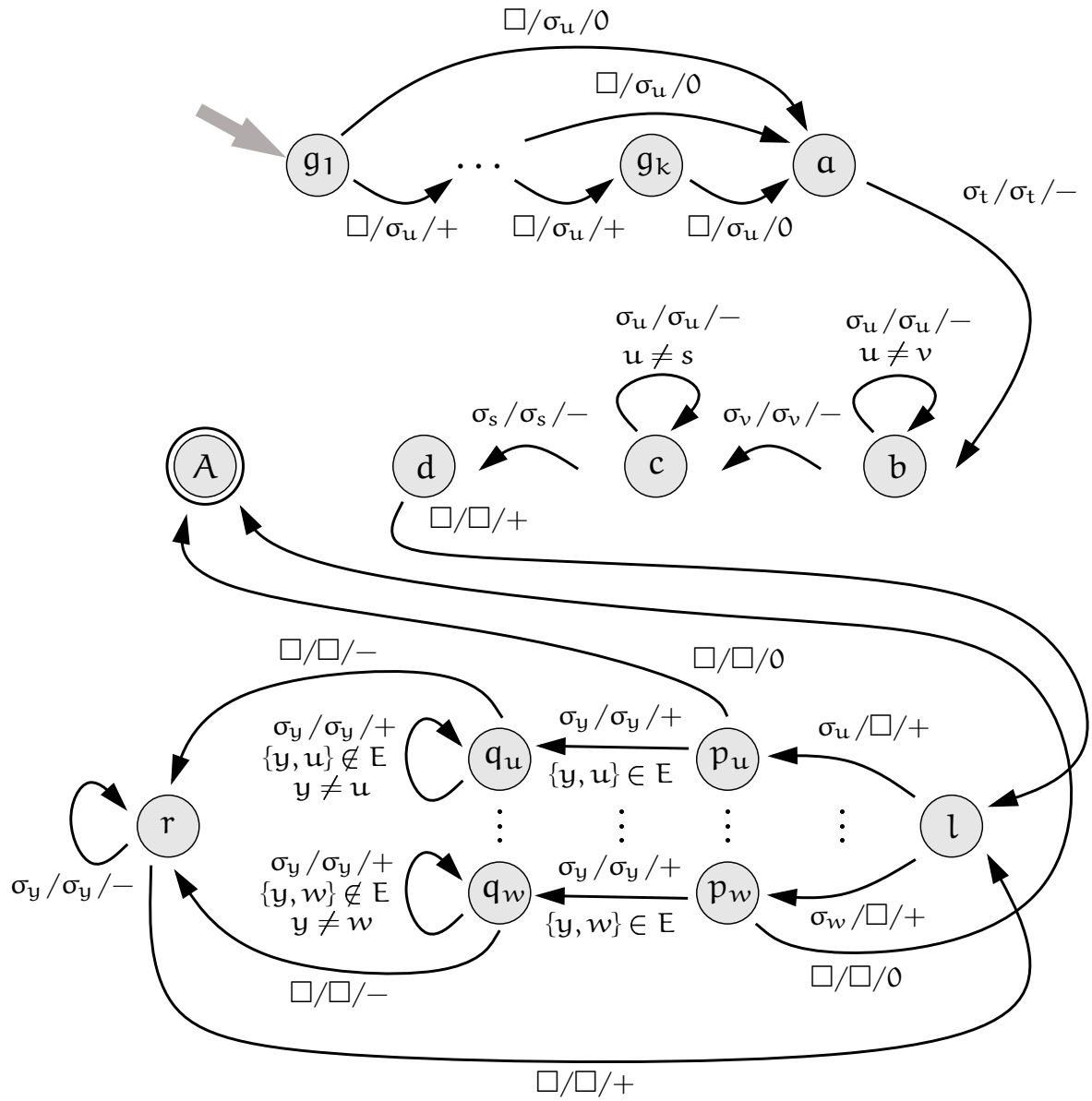


Figure 51: *A schematic description of the Turing machine defined in the proof of Theorem 5.5. The transition arrows are labeled by, in order, symbol under head, symbol to write, and head movement. To increase readability the final rejecting state and all transitions to it have been omitted.*

a diamond. As the naming of the vertices suggests, we associate each of these paths with a vertex from G in a bijective manner: routing a path through v_j^i, for some $1 \leq j \leq n$, is interpreted as selecting v_j to be part of the independent set I to be constructed. The construction uses k such vertex choice diamonds, which are connected by identifying s^{i+1} and t^i, for all $1 \leq i < k$. Let us call the graph described so far G_{VC}, where VC stands for vertex choice.

Proposition 5.7 *Any chordless (s^1, t^k)-path of size ℓ in G_{VC} corresponds to a multiset of $\ell - k - 1$ vertices in G.* □

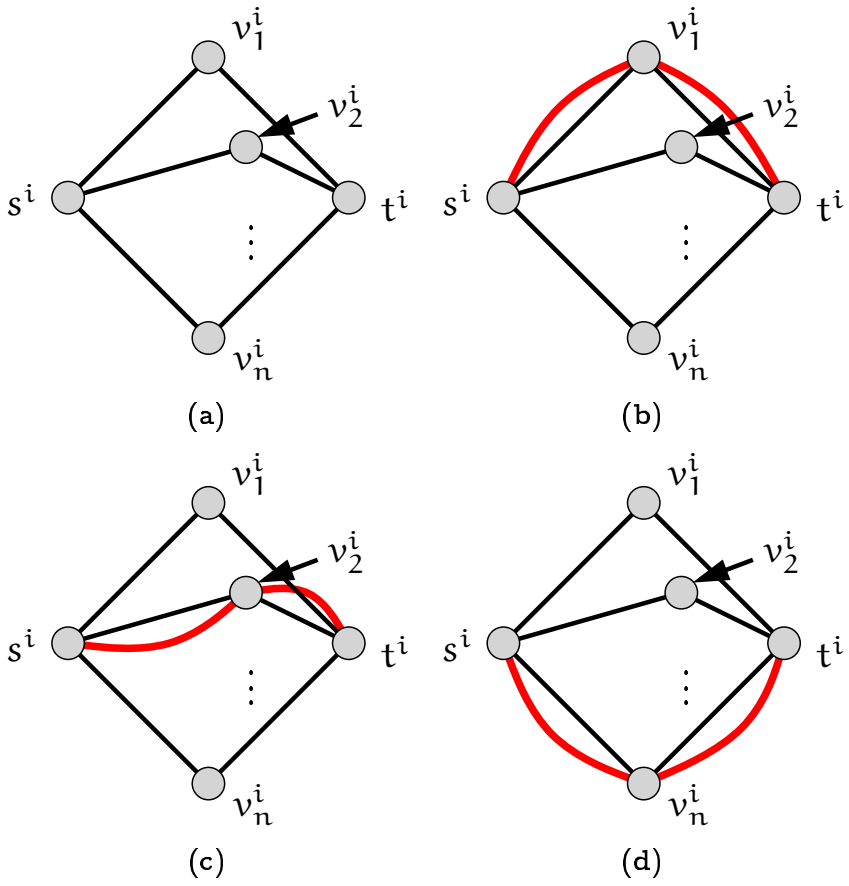


Figure 52: *A vertex choice diamond and three of its n chordless (s^i, t^i)-paths.*

The next step is to ensure that the vertices chosen by traversing G_{VC} on a chordless path correspond to an independent set in the original graph G. To accomplish this, we construct G' from two symmetric copies of G_{VC}. Denote the vertices in the first copy C of G_{VC} by s^i, v_j^i, and t^i, whereas the vertices in the second copy Γ of G_{VC} are referred

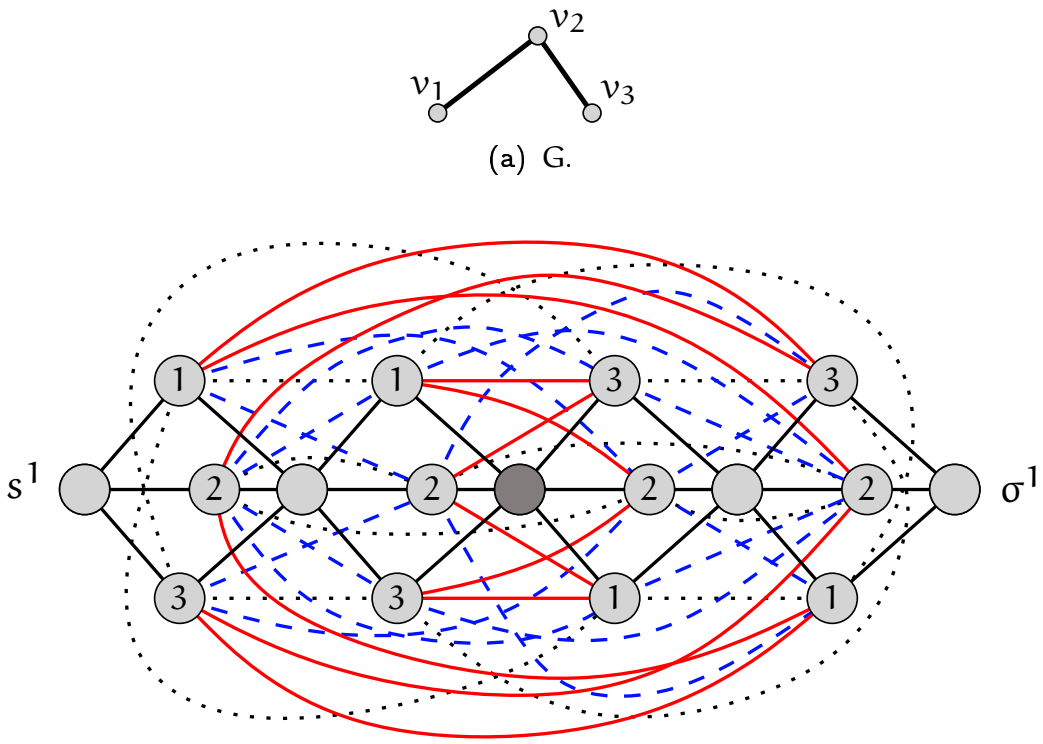
to as σ^i, φ^i_j, and τ^i, for $1 \leq i \leq k$ and $1 \leq j \leq n$. The graphs C and Γ are connected by identifying t^k and τ^k. The construction of G' is completed by adding a number of edges that encode the adjacency of G. An example is shown in Figure 53.

- There is an edge in G' between v^i_j and φ^i_ℓ, for all $1 \leq i \leq k$ and all $1 \leq j, \ell \leq n$ with $j \neq \ell$. Such an edge is called a *consistency edge*.
- For every edge $\{v_p, v_q\}$ in G, connect the vertex sets $\{v^i_p, \varphi^i_p\}$ and $\{v^j_q, \varphi^j_q\}$, for all $1 \leq i, j \leq k$ with $i \neq j$, by a complete bipartite subgraph in G'. These edges are called *independence edges*.
- Connect the vertex sets $\{v^i_\ell, \varphi^i_\ell\}$ and $\{v^j_\ell, \varphi^j_\ell\}$, for all $1 \leq i, j \leq k$ with $i \neq j$ and all $1 \leq \ell \leq n$, by a complete bipartite subgraph in G'. These edges are called *set edges*.

Lemma 5.8 *No chordless (s^1, σ^1)-path via t^k in G' uses a consistency edge or an independence edge or a set edge.*

Proof. Let P be a chordless (s^1, σ^1)-path via t^k in G'. Consider the initial part of P which traverses (part of) the first vertex choice diamond of C. By construction of G', exactly one of the vertices v^1_j, $1 \leq j \leq n$ is on P. (All of these vertices are neighbors of s^1.) Similarly, the final part of P contains exactly one of the vertices φ^1_ℓ, $1 \leq \ell \leq n$. If $\ell \neq j$, then the vertices v^1_j and φ^1_ℓ are connected by a consistency edge in G'. Hence, they together with s^1 and σ^1 induce an (s^1, σ^1)-path in G' that does not visit t^k. Such a path cannot be extended to a chordless path that visits t^k. (In fact, a chordless (s, t)-path cannot be extended in *any* way and at the same time remain a chordless (s, t)-path.) Thus, we conclude that $\ell = j$.

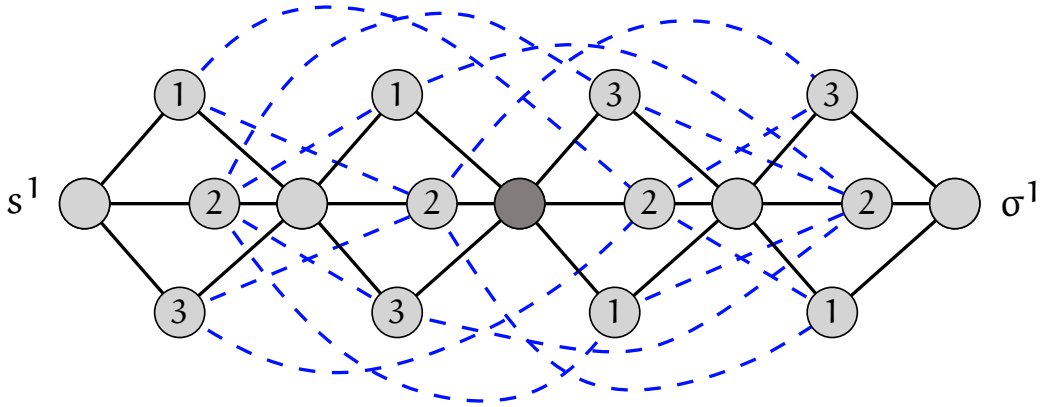
Furthermore, v^1_j and φ^1_j have the same neighbors along both independence and set edges by construction. Suppose that P continues from v^1_j via an independence (or set) edge to a vertex w. Clearly $w \neq t^k$ because t^k is not incident to any independence or set edge. As w is also a neighbor of φ^1_j, the vertices $\{s^1, v^1_j, w, \varphi^1_j, \sigma^1\} \subseteq V(P)$ induce an (s^1, σ^1)-path which does not visit t^k. Hence, P does not visit t^k, either, contrary to our assumption. Therefore, P cannot use any independence or set edge incident to v^1_j or φ^1_j.



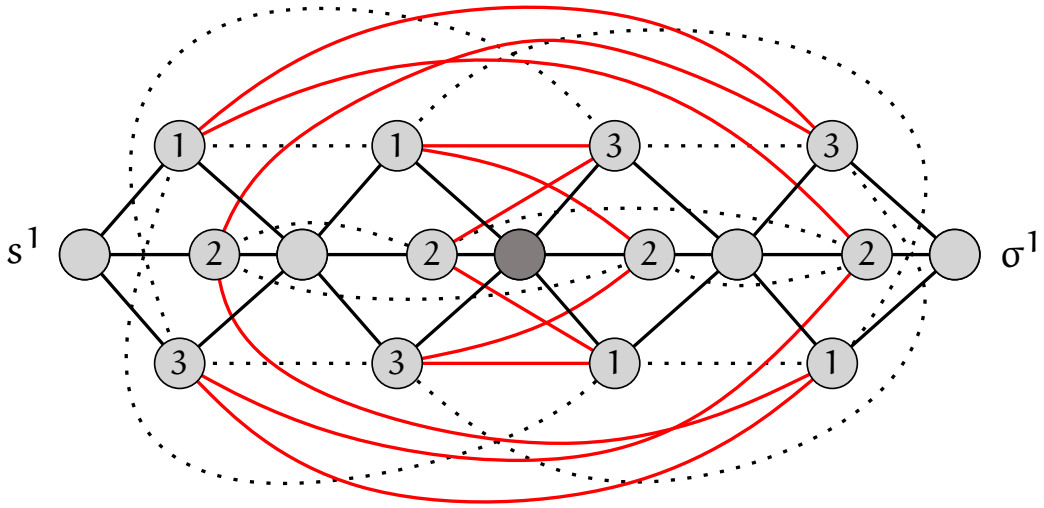
(a) G.

(b) G′.

Figure 53: *An example illustrating the construction of* G' *for* $k = 2$. *Consistency edges are shown by solid lines, independence edges by dashed lines, and set edges by dotted lines. The vertex labels in* G' *indicate the correspondence to the vertices from* G*: for example, the vertices labeled "*1*" correspond to* v_1. *The vertex* $t^k = \tau^k$ *is shaded dark.*



(a) G′ (independence edges only).



(b) G′ (without independence edges).

Figure 54: *The example from Figure 53 grouped into different types of edges.*

It remains to consider the consistency edges incident to v_j^1 and φ_j^1. Suppose P continues from v_j^1 on a consistency edge towards a vertex φ_ℓ^1, for some $1 \leq \ell \leq n$ with $\ell \neq j$. (All neighbors of v_j^1 along consistency edges are of this type.) Then the vertices $\{s^1, v_j^1, \varphi_\ell^1, \sigma^1\} \subseteq V(P)$ induce a chordless (s^1, σ^1)-path that does not visit t^k. Hence, P does not visit t^k, either, contrary to our assumption. Therefore, P cannot use any consistency edge incident to v_j^1 or φ_j^1.

In summary, the initial part of P goes from s^1 via v_j^1, for some $1 \leq j \leq n$, to $t^1 = s^2$, and the final part of P is completely symmetric: from τ^1 via φ_j^1 to σ^1. By induction on k, we can prove the following statement: The initial part of P is an (s^1, t^k)-path Q that visits all s^i in increasing order, for $1 \leq i \leq n$, without using any consistency, independence, or set edge, and the final part of P is a (τ^k, σ^1)-path that is completely symmetric to Q.

The analysis from above provides the base case of the induction for $k = 1$. At the same time, it also provides the induction step. As the initial and final section of the path are determined, we can remove the first vertex diamond and its symmetric copy from the graph and apply the induction hypothesis to the remaining $k - 1$ diamonds. □

Theorem 5.9 CP3V *is* $W[1]$*-hard with respect to* k.

Proof. Given an instance (G, k) of Independent Set, we construct the graph G' as described above. The graph G' contains $2k(n + 1) + 1$ vertices. To compute the number of edges in G' note that there are $4kn$ edges in the $2k$ vertex choice diamonds, plus $kn(n-1)$ consistency edges, $4mk(k-1)$ independence edges, and $2nk(k-1)$ set edges, where $n := |V(G)|$ and $m := |E(G)|$. Hence, G' can be constructed from G in time and space polynomial in both n and k.

Let P be a chordless (s^1, σ^1)-path via t^k of size at most $4k + 1$ in G'. By Lemma 5.8, P has a very special form: in particular, its size is exactly $4k + 1$, and it visits exactly one vertex $v_{j_i}^i$ from each of the vertex choice diamonds with $1 \leq i \leq k$. Let $I := \{v_{j_i} \in V(G) \mid v_{j_i}^i \in V(P) \text{ for } 1 \leq i \leq k\}$. Suppose that $v_{j_i} = v_{j_\ell}$ for some $1 \leq i, \ell \leq k$ with $i \neq \ell$. As $v_{j_i}^i$ and $v_{j_\ell}^\ell$ are connected by a set edge in G' that is not in P by Lemma 5.8, this set edge forms a chord of P, in contradiction to our assumption that P is chordless. Therefore, the vertices $v_{j_i}^i$ visited by

P correspond to mutually distinct vertices in G, that is, together with Proposition 5.7 it follows $|I| = 2k + 1 - k - 1 = k$.

Furthermore, we claim that I is an independent set in G. Suppose that for two vertices $v^i_{j_i}$ and $v^\ell_{j_\ell}$ on P, $1 \leq i, \ell \leq k$ and $i \neq \ell$, the corresponding vertices v_{j_i} and v_{j_ℓ} are neighbors in G. Then by construction $v^i_{j_i}$ and $v^\ell_{j_\ell}$ are connected by an independence edge in G'. This edge is not in P by Lemma 5.8, that is, it forms a chord of P, in contradiction to our assumption that P is chordless. Therefore, no two vertices in I are adjacent in G.

Conversely, for any independent set $I = \{v_1, v_2, \ldots, v_k, \ldots\}$ of size at least k in G there is a chordless (s^1, σ^1)-path P via t^k of size $4k + 1$ in G': in the i-th vertex choice diamonds, P visits v^i_i and φ^i_i, for $1 \leq i \leq k$.

Therefore, we have a parameterized reduction from an independent set instance (G, k) to a Cp3v instance $(G', 4k + 1)$, establishing $W[1]$-hardness of Cp3v. □

In general, $W[1]$-hardness does not (at least, is not known to) imply Np-hardness of the corresponding unparameterized problem. But in this particular case, the reduction can be extended easily.

Corollary 5.10 *It is* Np*-complete to decide whether there exists any chordless* (s, v, t)*-path for three given vertices* s*,* v*, and* t *of an undirected graph.*

Proof. The problem is clearly in Np. Regarding the hardness proof observe that the reduction described in Theorem 5.9 constructs the graph G' in time and space polynomial in both n and k and $k \leq n$. Moreover, the size of any chordless (s^1, σ^1)-path via t^k in G' is exactly $4k + 1$. That is, in order to solve the independent set problem in G, it is sufficient to decide on the existence of any chordless (s^1, σ^1)-path via t^k in G'. □

Analogously, the problem remains $W[1]$-complete, if we impose additional constraints on the chordless path under consideration, such as an exact size or parity.

Corollary 5.11 *It is* $W[1]$*-complete (w.r.t.* k*) to decide whether there exists a chordless* (s, v, t)*-path of size exactly* k *for three given vertices* s*,* v*, and* t *of an undirected graph and a positive integer* k*.* □

5.3 Chordless Cycles

In this section we discuss the relationship between the problem CP3V of deciding on the existence of a chordless path through three given vertices and the problem MCP2V ("Many chordless paths through two vertices") of deciding on the existence of a disjoint union of ℓ chordless (s,t)-paths, for some $\ell \in \mathbb{N}$.

Problem 5.12 (Many Chordless (s,t)-Paths (MCP2V))
Given an undirected graph $G = (V,E)$, positive integers k and ℓ, and two distinct vertices $s,t \in V$, is there a set $U \subseteq V$ of at most k vertices such that the subgraph induced by U in G is a disjoint union of ℓ chordless (s,t)-paths?

A subgraph H of G is a *disjoint union* of chordless (s,t)-paths, if in $H \setminus \{s,t\}$ each component is

- either an isolated vertex adjacent to both s and t in G;
- or a path $(p_1,\ldots,p_r)$, $r \geq 2$, for which
  - p_1 is adjacent to s but not to t in G;
  - p_r is adjacent to t but not to s in G;
  - every p_i, $2 \leq i < r$, is adjacent to neither s nor t in G.

We show that CP3V and MCP2V are equivalent under both parameterized reductions and classical (Karp–)reductions. This implies that both problems are complete for both $W[1]$ and NP.

Theorem 5.13 *Many Chordless (s,t)-Paths is $W[1]$-hard with respect to k, for any $\ell \geq 2$.*

Proof. For $\ell = 2$ we face the problem 2CP2V: Is there a chordless cycle of size at most k through s and t?

The proof is by reduction from CP3V. Consider a CP3V-instance (G,s,v,t,k). Construct a graph G' from G by adding a new vertex c that is adjacent to s and t only. Any chordless cycle of size at most $k+1$ through c and v in G' corresponds to a chordless (s,v,t)-path of size at most k in G and vice versa.

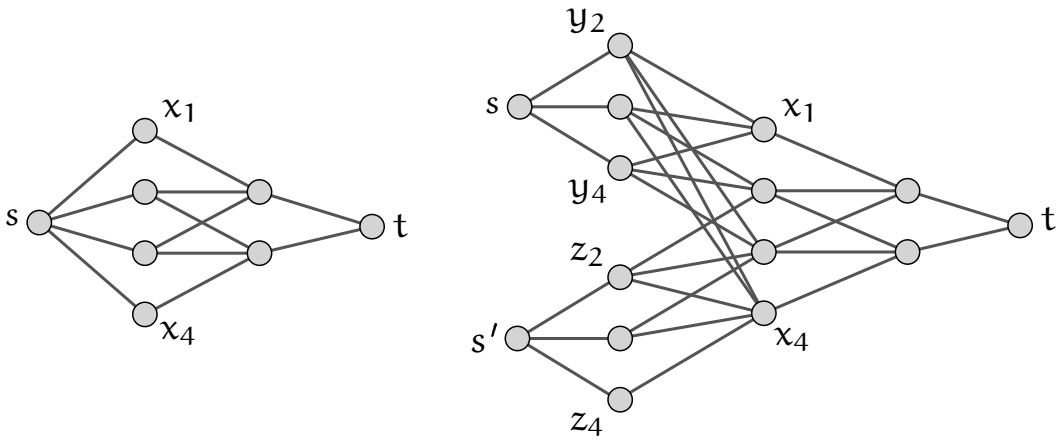
For $\ell > 2$ we add $\ell - 2$ additional vertices to G', each of them adjacent to c and v only. Then any set of at most $k + \ell - 1$ vertices in G' that form a disjoint union of ℓ chordless (c, v)-paths corresponds to a chordless (s, v, t)-path of size at most k in G and vice versa. □

The above reduction together with Corollary 5.10 is easily seen to prove NP-completeness of MCP2V. But this result is already known [35]. In fact, it provides an alternative way to prove Corollary 5.10, as outlined below.

Proof. (of Corollary 5.10 (alternative)) The problem is clearly in NP. The hardness is proven by reduction from 2CP2V. Consider an instance (G, s, t, k) of 2CP2V. Let $N_G(s) = \{x_1, \ldots, x_d\}$ with $d = \deg_G(s)$.

If $t \in N_G(s)$, then simply remove the edge $\{s, t\}$. A chordless (s, t)-path in the resulting graph corresponds to a chordless cycle through s and t in G and vice versa.

If $t \notin N_G(s)$, then construct a graph G' from G by adding a new vertex s' and new vertices $y_2, \ldots, y_d$ and $z_2, \ldots, z_d$. Remove all edges incident to s and connect s to all of $y_2, \ldots, y_d$ instead. Connect y_i to x_j, for all $2 \leq i \leq d$ and $1 \leq j \leq d$ with $i \neq j$. Finally, connect s' to all of $z_2, \ldots, z_d$ and connect z_i to x_j, for all $2 \leq i \leq j \leq d$. An example is depicted in Figure 55. Clearly, G' can be constructed from G in $O(|V(G)|^2)$ time.



(a) G. (b) G′.

Figure 55: *An example illustrating the construction of* G'.

We claim that any chordless (s, t, s')-path of size $k + 3$ in G' corresponds to a chordless cycle of size k through s and t in G and vice versa.

Indeed, any chordless cycle C through s and t in G passes through exactly two neighbors x_i and x_j of s, where $1 \leq i < j \leq d$. Such a cycle corresponds to a chordless path $(s, y_j, x_i, \ldots, t, \ldots, x_j, z_j, s')$-path in G' where the dotted part is $C \setminus s$. Conversely, any (s, t, s')-path in G' follows the pattern $(s, y_j, x_i, \ldots, t, \ldots, x_j, z_j, s')$, for $1 \leq i < j \leq d$, by construction of G' and corresponds to a chordless cycle $(s, x_i, \ldots, t, \ldots, x_j, s)$ in G. This proves the claim and the theorem. □

Note that the reduction described above is parametric. As a consequence, 2CP2V is in $W[1]$ and hence by Theorem 5.13 $W[1]$-complete. In order to extend this statement to the Many Chordless (s, t)-Paths Problem, we can once again use a reduction to SNTMC.

Theorem 5.14 *Many Chordless (s, t)-Paths is $W[1]$-complete with respect to k, for any $\ell \geq 2$.*

Proof. The hardness part was shown in Theorem 5.13; it remains to prove membership, which is done by reduction to SNTMC.

Consider an instance (G, s, t, k, ℓ) of MCP2V, where $G = (V, E)$ is an undirected graph, $s, t \in V$, and k and ℓ are positive integers. For the sake of brevity, we do not explicitly define a Turing machine as in Theorem 5.5 here. Instead we only sketch how such a machine could be constructed along the lines of Theorem 5.5 such that the number of steps in its computation can be bounded in terms of k. In the following, we identify the vertices of G with their corresponding symbols from the alphabet of the Turing machine.

First Phase. As the solution set $U \subset V(G)$ induces a disjoint union of (s, t)-paths, we can enumerate the vertices of U as a single path from s to either s or t (depending on whether ℓ is even or odd, respectively), where only s and t appear multiple times. The machine first "guesses" the at most k vertices of U in this particular order, thereby writing at most $k + \ell - 1$ vertices onto the tape.

Second Phase. The machine checks in $O(k)$ transitions whether the sequence of vertices on the tape starts with s, ends with s or t (depending on whether ℓ is even or odd, respectively), and whether s and t appear alternately in this sequence and both together $\ell + 1$ times.

Third Phase. The machine checks in $O(k)$ transitions whether every vertex on the tape that is adjacent to s in G is also adjacent to s on the tape. Similarly, it checks whether every vertex on the tape that is

adjacent to t in G is also adjacent to t on the tape.

Fourth Phase. The machine erases the vertices from the tape one after the other. For each removed vertex v, the vertex v' immediately following it on the tape has to be adjacent to v. If $v \in \{s, t\}$, then there is nothing more to do. Otherwise, for every remaining vertex $w \notin \{s, t, v'\}$ on the tape it has to be verified that $w \neq v$ and that w is not adjacent to v in G. This phase can be implemented using $O(k^2)$ transitions.

In summary, we have described a Turing machine that can be constructed for any specific instance (G, s, t, k, ℓ) of MCP2V in time polynomial in all of $|V(G)|$, k, and ℓ and whose computation reaches a final accepting state in $O(k^2)$ transitions, if and only if (G, s, t, k, ℓ) is solvable. □

5.4 Directed Graphs

The notion of chordless paths generalizes in a straightforward manner to directed graphs. A path P in a directed graph G is *chordless* if P is the directed subgraph induced by $V(P)$ in G.

Problem 5.15 (Directed Chordless (s,t)-Path (DCP)) *For a directed graph $G = (V, E)$, a positive integer k, and two distinct vertices $s, t \in V$, is there a chordless directed (s,t)-path of size at most k in G?*

Fellows et al. [36] showed that DCP is NP-complete even if restricted to planar digraphs. Our constructions described above can easily be adapted to deal with the directed setting as well.

Theorem 5.16 DCP *is $W[1]$-complete with respect to k.*

Proof. In the Turing machine of Theorem 5.5 replace all conditions that require the existence of an edge by corresponding conditions requiring the presence of a directed edge. Similarly, all conditions requiring the absence of an edge are replaced by corresponding conditions disallowing both directed edges.

The construction described in Theorem 5.9 is modified as follows. In the vertex choice diamonds direct all edges from s^i to v^i_j and from v^i_j to t^i, for all $1 \leq i \leq k$ and all $1 \leq j \leq n$. In the symmetric copy,

direct all edges from τ^i to φ^i_j and from φ^i_j to σ^i, for all $1 \leq i \leq k$ and all $1 \leq j \leq n$. These orientations induce a linear[1] order $(s^1, \ldots, t^k = \tau^k, \ldots, \sigma^1)$ on $V(G')$.

The remaining edges, that is, the consistency, independence, and set edges all are oriented from the vertex that is greater with respect to this linear order to the smaller vertex. It is easy to verify that no chordless directed (s^1, σ^1)-path can use a consistency, independence, or set edge. In fact, the orientation is chosen such that any chordless directed (s^1, σ^1)-path in G' passes through $t^k = \tau^k$, although this is not required by definition, in contrast to CP3V. □

As a consequence, also the following problem is $W[1]$-complete with respect to k. (Just add a single directed edge (σ^1, s^1) to the construction described in Theorem 5.16.)

Problem 5.17 (Directed Chordless Cycle) *For a directed graph $G = (V, E)$, a positive integer k, and a vertex $s \in V$, is there a chordless directed cycle of size at most k through s in G?*

Note that both problems are polynomial if the path or cycle is not required to be chordless, as the maximum number k of vertex-disjoint directed (s, t)-paths can be computed in $O(k|E|)$ time using flow techniques [38]. However, deciding whether there exist a directed (s_1, t_1)-path and a directed (s_2, t_2)-path that are vertex-disjoint is NP-complete, even for $t_1 = s_2$ and $t_2 = s_1$ [39].

Also, if the definition of chordless is relaxed to allow "back-cutting" arcs within each path, DCP restricted to planar graphs is polynomial, even for an arbitrary but fixed number of chordless (s, t)-paths [60]. The existence of such arcs is the crucial difference between the directed and the undirected problem: in an undirected (s, t)-path P every edge joining two vertices that are non-adjacent along P can be used as a shortcut. That is, the presence of any (s, t)-path implies the existence of a chordless (s, t)-path. However, we will show below that admitting back-cutting arcs does not change the parametric complexity of the problem for general graphs.

[1] Strictly speaking, the orientations define a partial order. It can be extended to a linear order by defining the order of the vertices v^i_j (similarly φ^i_j) within the same vertex choice diamond in an arbitrary manner.

Definition 5.18 *An* (s,t)*-path* P *in a graph* $G = (V,E)$ *is called* **weakly chordless** *if and only if* P *is a shortest* (s,t)*-path in* $G[V(P)]$.

Observe that there is no difference between chordless and weakly chordless in undirected graphs. But, in contrast to DCP, the presence of a directed weakly chordless (s,t)-path (or chordless cycle through s) can be decided in linear time by a breadth-first search. However, the obvious generalization to several paths defined below is again $W[1]$-complete, already for two paths.

Problem 5.19 (Many Weakly Chordless (s,t)-Paths) *For a directed graph* $G = (V,E)$*, positive integers* k *and* ℓ*, and two distinct vertices* $s,t \in V$*, is there a set* $U \subseteq V$ *of at most* k *vertices such that* $G[U]$ *is a disjoint union of* ℓ *weakly chordless* (s,t)*-paths?*

Theorem 5.20 *Many Weakly Chordless* (s,t)*-Paths is* $W[1]$*-complete with respect to* k*, for any* $\ell \geq 2$.

Proof. It is clear how to adapt the Turing machine construction of Theorem 5.5 to establish membership in $W[1]$.

For the hardness proof, consider the case $\ell = 2$. The construction described in Theorem 5.9 is modified as follows. First, add a directed edge from σ^1 to s^1, and let $s = t^k = \tau^k$ and $t = s^1$. In the vertex choice diamonds direct all edges from t^i to v^i_j and from v^i_j to s^i, for all $1 \leq i \leq k$ and all $1 \leq j \leq n$. Likewise, in the symmetric copy direct all edges from τ^i to φ^i_j and from φ^i_j to σ^i, for all $1 \leq i \leq k$ and all $1 \leq j \leq n$. Remove all independence and set edges within the same diamond chain, such that all remaining independence or set edges are between v^i_j and φ^p_q, for some $1 \leq i,p \leq k$ and $1 \leq j,q \leq n$. Direct those edges from v^i_j towards φ^p_q. See Figure 56 for an example.

Consider (t^k,s^1)-paths P and Q in G' such that $G[V(P) \cup V(Q)]$ is a disjoint union of weakly chordless (t^k,s^1)-paths. The way the edges are directed, one of the paths, say, P comes via the vertex choice diamonds and visits the vertices $t^k, s^k, t^{k-1}, \ldots t^1, s^1$, in order. On the other hand, Q traverses the symmetric copy and visits the vertices $\tau^k, \sigma^k, \tau^{k-1}, \ldots \tau^1, \sigma^1$, in order, before finally reaching s^1 via the added edge. Because there must not be any edge between P and Q, we can argue as in Theorem 5.9 that the vertices v^i_j and φ^p_q visited by P and Q, respectively, correspond to an independent set of size at most k in G.

For $\ell > 2$, we add $\ell - 2$ additional vertices to G' each of which is connected to s^1 (directed towards s^1) and t^k (directed from t^k) only. □

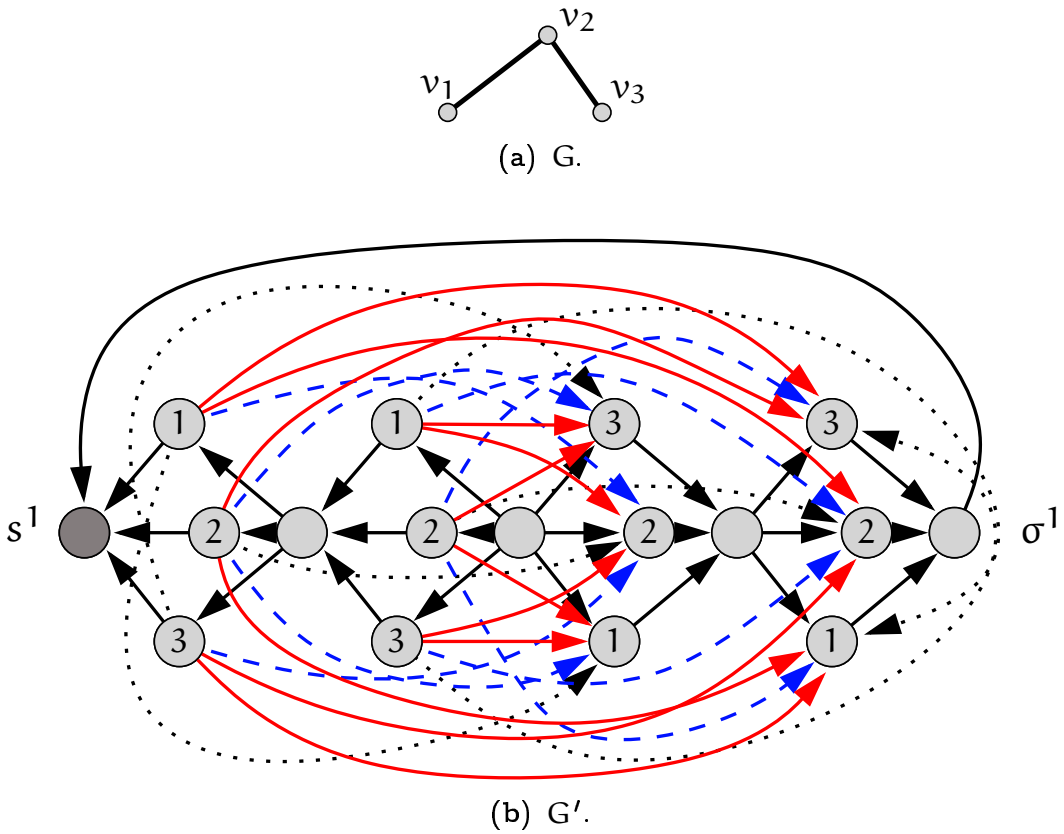


Figure 56: *An example illustrating the construction of* G' *for* $k = 2$. *Consistency edges are shown by solid lines, independence edges by dashed lines, and set edges by dotted lines. The vertex labels in* G' *indicate the correspondence to the vertices from* G*: for example, the vertices labeled "*1*" correspond to* v_1. *The vertex* $t = s^1$ *is shaded dark.*

Bibliography

[1] Pankaj K. Agarwal, Noga Alon, Boris Aronov, and Subhash Suri, Can Visibility Graphs be Represented Compactly?, *Discrete Comput. Geom.* **12** (1994), 347–365.

[2] Oswin Aichholzer, Michael Hoffmann, Bettina Speckmann, and Csaba D. Tóth, Degree Bounds for Constrained Pseudo-Triangulations, in: *Proc. 15th Canad. Conf. Comput. Geom.*, 2003, 155–158.

[3] Oswin Aichholzer, Clemens Huemer, and Hannes Krasser, Triangulations Without Pointed Spanning Trees, in: *Abstracts 20th European Workshop Comput. Geom.*, 2004, 221–224.

[4] Noga Alon, Sridhar Rajagopalan, and Subhash Suri, Long Non-Crossing Configurations in the Plane, in: *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, 1993, 257–263.

[5] Helmut Alt, Ulrich Fuchs, and Klaus Kriegel, On the Number of Simple Cycles in Planar Graphs, *Combinatorics, Probability and Computing* **8**, 5 (1999), 397–405.

[6] Esther M. Arkin, Martin Held, Joseph S.B. Mitchell, and Steven S. Skiena, Hamiltonian Triangulations for Fast Rendering, *Visual Comput.* **12**, 9 (1996), 429–444.

[7] Sanjeev Arora, Polynomial Time Approximation Schemes for Euclidean Traveling Salesman and other Geometric Problems, *J. ACM* **45**, 5 (1998), 753–782.

[8] David Avis and David Rappaport, Computing Monotone Simple Circuits in the Plane, in: *Computational Morphology* (God-

fried T. Toussaint, ed.), North-Holland, Amsterdam, Netherlands, 1988, 13–23.

[9] Alexander Barvinok, Sándor P. Fekete, David S. Johnson, Arie Tamir, Gerhard J. Woeginger, and Russ Woodroofe, The Geometric Maximum Traveling Salesman Problem, *J. ACM* **50**, 5 (2003), 641–664.

[10] Cristina Bazgan, Schémas d'Approximation et Complexité Paramétrée, Rapport du stage (DEA), Université Paris Sud, 1995.

[11] Claude Berge, Färbung von Graphen deren sämtliche beziehungsweise deren ungerade Kreise starr sind (Zusammenfassung)., *Wiss. Z. Martin Luther Univ. Halle Wittenberg Math. Naturwiss. Reihe* 114–115.

[12] Sergei Bespamyatnikh, Computing Homotopic Shortest Paths in the Plane, *J. Algorithms* **49**, 2 (2003), 284–303.

[13] Dan Bienstock, On the Complexity of Testing for Odd Holes and Induces Odd Paths, *Discrete Math.* **90**, 1 (1991), 85–92.

[14] Dan Bienstock, Corrigendum to: On the Complexity of Testing for Odd Holes and Induces Odd Paths, *Discrete Math.* **102**, 1 (1992), 109.

[15] Prosenjit Bose, Michael E. Houle, and Godfried T. Toussaint, Every Set of Disjoint Line Segments Admits a Binary Tree, *Discrete Comput. Geom.* **26** (2001), 387–410.

[16] Prosenjit Bose and Godfried T. Toussaint, Growing a Tree from its Branches, *J. Algorithms* **19**, 1 (1995), 86–103.

[17] Liming Cai, Jianer Chen, Rodney G. Downey, and Michael R. Fellows, On the Parameterized Complexity of Short Computation and Factorization, *Arch. Math. Logic* **36**, 4–5 (1997), 321–337.

[18] Jakub Černý, Zdeněk Dvořák, Vít Jelínek, and Jan Kára, Noncrossing Hamiltonian Paths in Geometric Graphs, in: *Graph Drawing* (Giuseppe Liotta, ed.), volume 2912 of *Lecture Notes Comput. Sci.*, Springer-Verlag, 2004, 86–97.

[19] Marco Cesati, Perfect Code is W[1]-complete, *Inform. Process. Lett.* **81**, 3 (2002), 163–168.

[20] Marco Cesati, The Turing Way to Parameterized Complexity, *J. Comput. Syst. Sci.* **67**, 4 (2003), 654–685.

[21] Marco Cesati and Luca Trevisan, On the Efficiency of Polynomial Time Approximation Schemes, *Inform. Process. Lett.* **64**, 4 (1997), 165–171.

[22] Maria Chudnovsky, Gérard Cornuéjols, Xinming Liu, Paul D. Seymour, and Kristina Vušković, Recognizing Berge Graphs, *Combinatorica* **25**, 2 (2005), 143–187.

[23] Maria Chudnovsky, Neil Robertson, Paul D. Seymour, and Robin Thomas, The Strong Perfect Graph Theorem, `http://www.math.gatech.edu/~thomas/spgc.ps.gz`, 2003. Manuscript.

[24] Joe Csima and Tom A. Ralston, Crossing-free Hamiltonian Paths in Euclidean Spaces, *Ars Combin.* **18** (1984), 87–97.

[25] Erik D. Demaine and Joseph O'Rourke, Open Problems from CCCG'99, Technical Report 066, Dept. Comput. Sci., Smith College, Northampton, MA, 2000.

[26] Michael B. Dillencourt, A Non-Hamiltonian, Nondegenerate Delaunay Triangulation, *Inform. Process. Lett.* **25** (1987), 149–151.

[27] Michael B. Dillencourt, Hamiltonian Cycles in Planar Triangulations with no Separating Triangles, *J. Graph Theory* **14**, 1 (1990), 31–49.

[28] Michael B. Dillencourt, Finding Hamiltonian Cycles in Delaunay Triangulations is NP-complete, *Discrete Appl. Math.* **64** (1996), 207–217.

[29] Rodney G. Downey and Michael R. Fellows, Fixed-parameter Tractability and Completeness II: On Completeness for W[1], *Theoret. Comput. Sci.* **141** (1995), 109–131.

[30] Rodney G. Downey and Michael R. Fellows, *Parameterized Complexity*, Monographs in Computer Science, Springer-Verlag, 1999.

[31] Rodney G. Downey, Michael R. Fellows, Bruce Kapron, Michael T. Hallett, and H. Todd Wareham, Parameterized Complexity and Some Problems in Logic and Linguistics, in: *Proc. 2nd Workshop on Structural Complexity and Recursion-theoretic methods in*

Logic-Programming, volume 813 of *Lecture Notes Comput. Sci.*, Springer-Verlag, 1994, 89–101.

[32] Alon Efrat, Stephen G. Kobourov, and Anna Lubiw, Computing Homotopic Shortest Paths Efficiently, in: *Proc. Annu. European Sympos. Algorithms*, number 2461 in Lecture Notes Comput. Sci., Springer-Verlag, 2003, 411–423.

[33] Hazel Everett, Chinh T. Hoang, Kyriakos Kilakos, and Marc Noy, Planar Segment Visibility Graphs, *Comput. Geom. Theory Appl.* **16** (2000), 235–243.

[34] Istvan Fary, On Straight Line Representation of Planar Graphs, *Acta Sci. Math. Szeged* **11** (1948), 229–233.

[35] Michael R. Fellows, The Robertson-Seymour Theorems: A Survey of Applications, in: *Proc. AMS-IMS-SIAM Joint Summer Research Conf.* (Bruce Richter, ed.), volume 89 of *Contemporary Mathematics*, American Mathematical Society, Providence, RI, 1989, 1–18.

[36] Michael R. Fellows, Jan Kratochvíl, Martin Middendorf, and Frank Pfeiffer, The Complexity of Induced Minors and Related Problems, *Algorithmica* **13** (1995), 266–282.

[37] Merrill M. Flood, The Traveling-Salesman Problem, *Oper. Res.* **4** (1956), 61–75.

[38] Lester R. Ford and Delbert R. Fulkerson, Maximal Flow through a Network, *Canad. J. Math.* **8** (1956), 399–404.

[39] Steve Fortune, John E. Hopcroft, and James Wyllie, The Directed Subgraph Homeomorphism Problem, *Theoret. Comput. Sci.* **10** (1980), 111–121.

[40] Alfredo García, Marc Noy, and Javier Tejel, Lower Bounds on the Number of Crossing-Free Subgraphs of K_N, *Comput. Geom. Theory Appl.* **16** (2000), 211–221.

[41] Michael R. Garey and David S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, New York, NY, 1979.

[42] Subir K. Ghosh and David M. Mount, An Output-Sensitive Algorithm for Computing Visibility Graphs, *SIAM J. Comput.* **20** (1991), 888–910.

[43] Branko Grünbaum, Hamiltonian Polygons and Polyhedra, *Geombinatorics* **3**, 3 (1994), 83–89.

[44] Robert Haas, *Service Deployment in Programmable Networks*, Ph.D. thesis, ETH Zurich, Switzerland, 2003.

[45] Robert Haas and Michael Hoffmann, Chordless Paths Through Three Vertices, in: *Proc. Internat. Workshop on Parameterized and Exact Computation*, volume 3162 of *Lecture Notes Comput. Sci.*, Springer-Verlag, 2004, 25–36.

[46] John Hershberger, An Optimal Visibility Graph Algorithm for Triangulated Simple Polygons, *Algorithmica* **4** (1989), 141–155.

[47] John Hershberger and Subhash Suri, An Optimal Algorithm for Euclidean Shortest Paths in the Plane, *SIAM J. Comput.* **28**, 6 (1999), 2215–2256.

[48] Michael Hoffmann, Bettina Speckmann, and Csaba D. Tóth, Pointed Binary Encompassing Trees, in: *Proc. 9th Scand. Workshop Algorithm Theory*, volume 3111 of *Lecture Notes Comput. Sci.*, Springer-Verlag, 2004, 442–454.

[49] Michael Hoffmann and Csaba D. Tóth, Segment Endpoint Visibility Graphs are Hamiltonian, in: *Proc. 13th Canad. Conf. Comput. Geom.*, 2001, 109–112.

[50] Michael Hoffmann and Csaba D. Tóth, Alternating Paths through Disjoint Line Segments, in: *Abstracts 18th European Workshop Comput. Geom.*, 2002, 23–26.

[51] Michael Hoffmann and Csaba D. Tóth, Alternating Paths through Disjoint Line Segments, *Inform. Process. Lett.* **87**, 6 (2003), 287–294.

[52] Michael Hoffmann and Csaba D. Tóth, Segment Endpoint Visibility Graphs are Hamiltonian, *Comput. Geom. Theory Appl.* **26**, 1 (2003), 47–68.

[53] Ferran Hurtado, Godfried T. Toussaint, and Joan Trias, On Polyhedra Induced by Point Sets in Space, in: *Proc. 15th Canad. Conf. Comput. Geom.*, 2003, 107–110.

[54] Alon Itai, Christos H. Papadimitriou, and Jayme L. Szwarcfiter, Hamilton Paths in Grid Graphs, *SIAM J. Comput.* **11** (1982), 676–686.

[55] Ray A. Jarvis, On the Identification of the Convex Hull of a Finite Set of Points in the Plane, *Inform. Process. Lett.* **2**, 1 (1973), 18–21.

[56] Richard M. Karp, On the Complexity of Combinatorial Problems, *Networks* **5** (1975), 45–68.

[57] Der-Tsai Lee and Franco P. Preparata, Euclidean Shortest Paths in the Presence of Rectilinear Barriers, *Networks* **14** (1984), 393–410.

[58] George S. Lueker, Donald J. Rose, and Robert E. Tarjan, Algorithmic Aspects of Vertex Elimination in Graphs, *SIAM J. Comput.* **5** (1976), 266–283.

[59] Colin McDiarmid, Bruce Reed, Alexander Schrijver, and Bruce Shepherd, Non-Interfering Network Flows, in: *Proc. 3rd Scand. Workshop Algorithm Theory*, 1992, 245–257.

[60] Colin McDiarmid, Bruce Reed, Alexander Schrijver, and Bruce Shepherd, Induced Circuits in Planar Graphs, *J. Combin. Theory Ser. B* **60** (1994), 169–176.

[61] Andranik Mirzaian, Hamiltonian Triangulations and Circumscribing Polygons of Disjoint Line Segments, *Comput. Geom. Theory Appl.* **2**, 1 (1992), 15–30.

[62] Joseph S.B. Mitchell, Guillotine Subdivisions Approximate Polygonal Subdivisions: A Simple Polynomial-Time Approximation Scheme for Geometric TSP, k-MST, and Related Problems, *SIAM J. Comput.* **28** (1999), 1298–1309.

[63] Joseph S.B. Mitchell, Shortest Paths and Networks, in: *Handbook of Discrete and Computational Geometry* (Jacob E. Goodman and Joseph O'Rourke, eds.), CRC Press LLC, Boca Raton, FL, 2004, 607–641.

[64] Joseph S.B. Mitchell and Joseph O'Rourke, Computational geometry column 42, *Internat. J. Comput. Geom. Appl.* **11**, 5 (2001), 573–582. Also in *SIGACT News* 32(3):63-72 (2001), Issue 120.

[65] David E. Muller and Franco P. Preparata, Finding the Intersection of Two Convex Polyhedra, *Theoret. Comput. Sci.* **7** (1978), 217–236.

[66] Giri Narasimhan, On Hamiltonian Triangulations in Simple Polygons, *Internat. J. Comput. Geom. Appl.* **9**, 3 (1999), 261–275.

[67] Rolf Niedermeier, *Invitation to Fixed-Parameter Algorithms*, Habilitation thesis, Wilhelm-Schickard Institut für Informatik, Universität Tübingen, Germany, 2002.

[68] Stavros D. Nikolopoulos and Leonidas Palios, Hole and Antihole Detection in Graphs, in: *Proc. 15th ACM-SIAM Sympos. Discrete Algorithms*, 2004, 843–852.

[69] Joseph O'Rourke and Jennifer Rippel, Two Segment Classes with Hamiltonian Visibility Graphs, *Comput. Geom. Theory Appl.* **4** (1994), 209–218.

[70] Joseph O'Rourke and Subhash Suri, Polygons, in: *Handbook of Discrete and Computational Geometry* (Jacob E. Goodman and Joseph O'Rourke, eds.), CRC Press LLC, Boca Raton, FL, 2004, 583–606.

[71] János Pach and Eduardo Rivera-Campo, On Circumscribing Polygons for Line Segments, *Comput. Geom. Theory Appl.* **10** (1998), 121–124.

[72] Christos H. Papadimitriou, The Euclidean Traveling Salesman Problem is NP-complete, *Theoret. Comput. Sci.* **4** (1977), 237–244.

[73] Micha Perles, DIMACS Workshop on Geometric Graph Theory, 2002.

[74] Michel Pocchiola and Gert Vegter, Topologically Sweeping Visibility Complexes via Pseudo-Triangulations, *Discrete Comput. Geom.* **16** (1996), 419–453.

[75] Louis V. Quintas and Fred Supnick, On some Properties of Shortest Hamiltonian Circuits, *Amer. Math. Monthly* **72** (1965), 977–980.

[76] Satish B. Rao and Warren D. Smith, Approximating geometrical graphs via "spanners" and "banyans", in: *Proc. 30th Annu. ACM Sympos. Theory Comput.*, 1998, 540–550.

[77] David Rappaport, Computing Simple Circuits from a Set of Line Segments is NP-complete, *SIAM J. Comput.* **18**, 6 (1989), 1128–1139.

[78] David Rappaport, The Visibility Graph of Congruent Discs is Hamiltonian, *Comput. Geom. Theory Appl.* **25**, 3 (2003), 257–265.

[79] David Rappaport, Hiroshi Imai, and Godfried T. Toussaint, Computing Simple Circuits from a Set of Line Segments, *Discrete Comput. Geom.* **5**, 3 (1990), 289–304.

[80] Neil Robertson and Paul D. Seymour, Graph Minors XIII. The Disjoint Paths Problem, *J. Combin. Theory Ser. B* **63** (1995), 65–110.

[81] Francisco Santos and Raimund Seidel, A Better Upper Bound on the Number of Triangulations of a Planar Point Set, *J. Combin. Theory Ser. A* **102**, 1 (2003), 186–193.

[82] Xiaojun Shen and Herbert Edelsbrunner, A Tight Lower Bound on the Size of Visibility Graphs, *Inform. Process. Lett.* **26** (1987), 61–64.

[83] Sherman K. Stein, Convex Maps, *Proc. Amer. Math. Soc.* **2**, 3 (1951), 464–466.

[84] Ernst Steinitz and Hans Rademacher, *Vorlesungen über die Theorie der Polyeder*, Julius Springer, Berlin, Germany, 1934.

[85] Robert E. Tarjan and Mihalis Yannakakis, Simple Linear-Time Algorithms to Test Chordality of Graphs, Test Acyclicity of Hypergraphs, and Selectively Reduce Acyclic Hypergraphs, *SIAM J. Comput.* **13** (1984), 566–579.

[86] Masatsugu Urabe and Mamoru Watanabe, On a Counterexample to a Conjecture of Mirzaian, *Comput. Geom. Theory Appl.* **2**, 1 (1992), 51–53.

[87] Jorge Urrutia, Algunos Problemas Abiertos (in Spanish), in: *Actas de los IX Encuentros de Geometría Computacional* (Narcís Coll and Joan Antoni Sellarès, eds.), 2001, 13–24.

[88] Jorge Urrutia, Open Problems in Computational Geometry, in: *Proc. Latin Amer. Sympos. Theoret. Informatics*, volume 2286 of *Lecture Notes Comput. Sci.*, Springer-Verlag, 2002, 4–11.

[89] Klaus Wagner, Bemerkungen zum Vierfarbenproblem (in German), *Jahresbericht der Deutschen Mathematiker-Vereinigung* **46** (1936), 26–32.

[90] Emo Welzl, Constructing the Visibility Graph for n Line Segments in $O(n^2)$ Time, *Inform. Process. Lett.* **20** (1985), 167–171.

[91] Hassler Whitney, A Theorem on Graphs, *Annu. Math.* **32** (1931), 378–390.

Curriculum Vitae

Michael Hoffmann

born on May 6, 1970 in Berlin, Germany

1980–1989 Highschool "Gymnasium Steglitz" in Berlin, Germany

1989–1996 Studies at Freie Universität Berlin, Germany

Major: Mathematics

Minor: Computer Science

Since 1996 Assistant

Institute for Theoretical Computer Science, ETH Zürich

Since 2000 PhD student at ETH Zürich