The Zigzag Path of a Pseudo-Triangulation

Oswin Aichholzer[1], Günter Rote[2*], Bettina Speckmann[3], and Ileana Streinu[4**]

[1] Institute for Software Technology, Graz University of Technology, oaich@ist.tugraz.at
[2] Institute of Computer Science, FU Berlin, rote@inf.fu-berlin.de
[3] Institute for Theoretical Computer Science, ETH Zürich, speckman@inf.ethz.ch
[4] Department of Computer Science, Smith College, streinu@cs.smith.edu

Abstract. We define the zigzag path of a pseudo-triangulation, a concept generalizing the path of a triangulation of a point set. The pseudo-triangulation zigzag path allows us to use divide-and-conquer type of approaches for suitable (i.e., decomposable) problems on pseudo-triangulations. For this we provide an algorithm that enumerates all pseudo-triangulation zigzag paths (of all pseudo-triangulations of a given point set with respect to a given line) in $O(n^2)$ time per path and $O(n^2)$ space, where n is the number of points. We illustrate applications of our scheme which include a novel algorithm to count the number of pseudo-triangulations of a point set.

1 Introduction

Pseudo-triangulations, unlike triangulations, only recently emerged as a promising data structure with a variety of applications. They were originally introduced in the context of visibility complexes [15] and ray shooting [8, 12], but in the last few years they also found application in robot arm motion planning [18], kinetic collision detection [1, 13], and guarding [17]. In particular the so-called minimum or pointed pseudo-triangulations introduced by Streinu [18] exhibit many fascinating properties that initiated a growing interest in their geometric and combinatorial nature.

There exist already several algorithms to enumerate pseudo-triangulations of sets of n points. Bespamyatnikh [5], extending his work on enumerating triangulations [6], defines a lexicographical order on pseudo-triangulations which he uses to enumerate pseudo-triangulations in $O(\log n)$ time per pseudo-triangulation. Brönnimann et al. [7] implemented an ad-hoc technique of Pocchiola based on a greedy strategy for generating edges of pseudo-triangulations. Unfortunately the time complexity of this algorithm is not known, but it requires $O(n^2)$ space. A third possibility is to apply some vertex enumeration algorithm to the polytope of pseudo-triangulations developed in [14, 16]. For example, Motzkin's double

* Research partly supported by the Deutsche Forschungsgemeinschaft (DFG) under grant RO 2338/2-1.
** Research supported by NSF grant CCR-0105507.

description method or the reverse-search technique of Avis and Fukuda [4] are two methods for vertex enumeration which have been implemented [3, 11].

We propose a different scheme for solving counting and optimization problems for pseudo-triangulations, inspired by an analogous approach developed for triangulations. The "path of a triangulation" was introduced by Aichholzer [2] in order to count the triangulations of a planar point set in a divide-and-conquer like manner. This concept can be used to attack any decomposable problem on triangulations. Dumitrescu et al. [9] provided an algorithm that enumerates all triangulation paths (of all triangulations of a given point set with respect to a given line) in $O(n^3 \log n)$ time per path and $O(n)$ space.

In this paper we describe a meaningful extension of the path concept to pseudo-triangulations. We first recall some definitions concerning pseudo-triangulations and also formalize the notion of a decomposable problem. In Section 4 and 5 we then develop the definition of the zigzag path of a pseudo-triangulation, which retains all of the useful properties of a triangulation path. Finally in Section 6 we show how to generate all pseudo-triangulation zigzag paths in $O(n^2)$ time per path (at the expense of $O(n^2)$ space and preprocessing time).

The path concept can be generalized to arbitrary (i.e., not necessarily pointed) pseudo-triangulations. However, in this extended abstract we concentrate on the results pertaining to pointed pseudo-triangulations. The extension to general pseudo-triangulations can be found in the journal version of this paper.

2 Pseudo-triangulations

We consider a simple planar polygon P and a point set $S \subseteq P$, $|S| = n$, which contains all vertices of P but may also contain additional *inner points*. We will assume throughout that S is in general position, i.e., it contains no three collinear points. We will refer to the pair (S, P) as a *point set S in a polygon P*, or shorter as *pointgon*. We denote the boundary of P by ∂P.

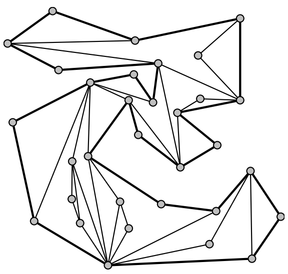

Fig. 1. A pointed pseudo-triangulation of a pointgon.

A *pseudo-triangle* is a planar polygon that has exactly three convex vertices, called *corners*, with internal angles less than π. A *pseudo-triangulation* T of a pointgon (S, P) is a partition of the interior of P into pseudo-triangles whose vertex set is exactly S (see Fig. 1) . A vertex p in a pseudo-triangulation T of (S, P) is *pointed* if there is one region incident to p (either a pseudo-triangle or the outer face) whose angle at p is greater than π. A pseudo-triangulation T of (S, P) is called *pointed* if each point $p \in S$ is pointed. A pseudo-triangulation for a point set S corresponds to the case where P is the convex hull of S.

Proposition 1 (Streinu [18]) *Every non-crossing pointed set of edges in a pointgon (S, P) can be extended to a pointed pseudo-triangulation of (S, P).*

3 Decomposable Problems and Divide-and-Conquer

We are interested in certain types of optimization or counting problems for the set of pseudo-triangulations for a point set S. We associate with each pseudo-triangulation a *zigzag path*, which decomposes the convex hull of S into several parts on which the problem can be solved recursively. Our approach can be summarized as follows:

1. Enumerate all zigzag paths α.
2. For each α:
3. Use α to split the problem into several pieces.
4. Solve each subproblem recursively.
5. Combine the solutions of the subproblems.
6. Combine the solutions for all zigzag paths into the solution for the original problem.

The main contribution of this paper is a proper definition of a zigzag path and an algorithm for enumerating zigzag paths, in order to carry out step 1 of this procedure.

The problem that we want to solve must have a certain decomposable structure in order to be amenable to this approach. This structure can be described by a commutative *semiring* $(H, \oplus, \otimes)$ with two associative and commutative operations $\oplus$ and $\otimes$ which satisfy the distributive law:

$$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$$

We assume that an "objective function" $f(T)$ for a pseudo-triangulation T can be computed as the $\otimes$-product of $f(t)$ for the individual pseudo-triangles $t \in T$, where $f(t) \in H$ is some function that is determined individually for each pseudo-triangle. We use the $\oplus$ operation to accumulate the values of all pseudo-triangulations into the quantity in which we are finally interested in. The task is to calculate

$$\tilde{f}(\mathcal{T}) := \bigoplus_{T \in \mathcal{T}} f(T) = \bigoplus_{T \in \mathcal{T}} \bigotimes_{t \in T} f(t) \tag{1}$$

over some set $\mathcal{T}$ of pseudo-triangulations T.

Now if we can generate all zigzag paths, then we can easily count the number of pseudo-triangulations as follow: $(H, \oplus, \otimes) = (\mathbb{N}, +, \cdot)$, with $f(t) \equiv 1$ for every pseudo-triangle t. We can also optimize various quantities over the set of pseudo-triangulations, for example the smallest angle, or the sum of the edge lengths. In the first case, we take $(H, \oplus, \otimes) = (\mathbb{R}, \max, \min)$, and $f(t) =$ the smallest angle in t. In the second case, we take $(H, \oplus, \otimes) = (\mathbb{R}, \min, +)$, and $f(t) =$ the perimeter of t. Here we count the length of the interior edges twice, but since the total length of the boundary edges is constant, this is equivalent to optimizing the total length.

As mentioned before, one can of course solve these problems, and more general optimization problems, by enumerating all pseudo-triangulations by one of the methods mentioned in the introduction, evaluating $f(T)$ for each pseudo-triangulation T, and taking the $\oplus$-sum. However, our divide-and-conquer procedure is usually several orders of magnitude faster than this trivial approach.

4 The Zigzag Path

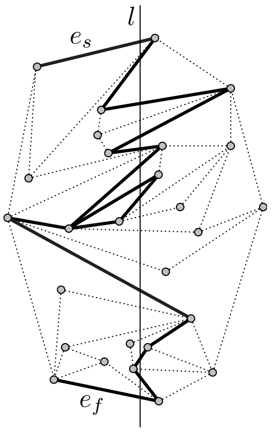

Fig. 2. The pseudo-triangulation zigzag path.

Assume that we are given a pseudo-triangulation T of a pointgon (S, P). We have to choose a *cut segment* l that connects two boundary points of P through the interior of P but avoids all points in S. For simplicity we will assume throughout the paper that l is vertical. The endpoints of l lie on two edges of P, the *start edge* e_s on the top and the *final edge* e_f on the bottom. Let $E = \{e_1 = e_s, e_2, \cdots, e_k = e_f\}$ be the set of edges of T that are crossed by l, ordered from top to bottom according to their intersection with l. Consider a pair of consecutive edges e_i and e_{i+1} in E. We say that the pair (e_i, e_{i+1}) *leans* to the left or to the right, respectively, according to the location of the intersection of the lines through e_i and e_{i+1} with respect to l. Since two edges of a common pseudo-triangle are never parallel, this direction is always well-defined. If (e_{i-1}, e_i) and (e_i, e_{i+1}) lean in different directions, the edge e_i is called a *signpost* (see Fig. 3.a–b). The starting and ending edges e_s and e_f are also considered to be signposts.

We define the *zigzag path* $\alpha_l(T)$ of a pseudo-triangulation T with respect to a cut segment l as follows: We remove all edges of E that are not signposts. Let P^* denote the resulting set of polygons, see Figure 3.c. We now construct $\alpha_l(T)$ by joining adjacent signposts along the boundary of their common face in P^*

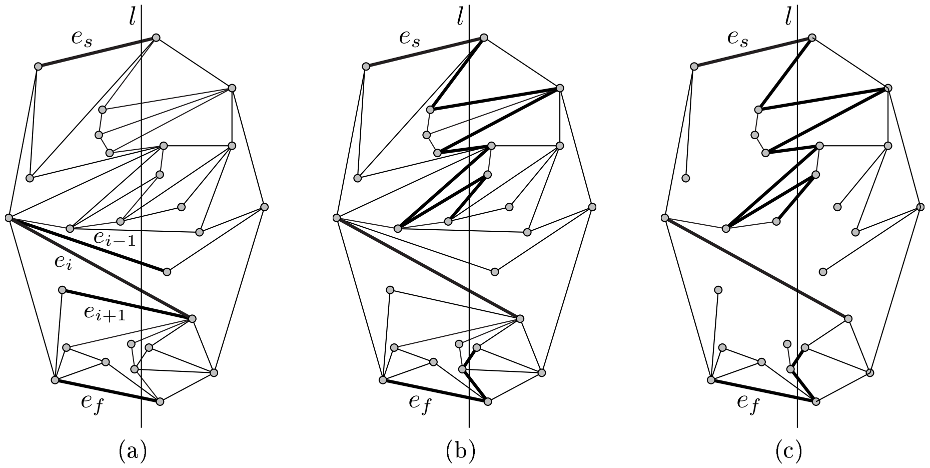

Fig. 3. Constructing the zigzag path of a pseudo-triangulation. (a) A pseudo-triangulation cut by a segment l — the pair (e_i, e_{i+1}) leans to the right. (b) The signposts. (c) Removing edges that are cut by l but are not signposts.

according to their lean, i.e., if two adjacent signposts lean to the left, then we connect them via the edges of their common polygon that lie to the left of l, see Fig. 2. Note that a vertex can appear on the path several times.

Before stating a couple of characteristic properties of the zigzag path, we introduce some terminology. Consider a pseudo-triangle $t \in T$ which is cut by l in precisely two edges e and f. Let l^+ denote the side of l to which e and f lean. Then the part of t that lies in l^+ is a pseudo-triangle. t has one corner v in l^+, which is called a *turning point*. v is connected to e and f via two x-monotone chains, whose vertices (excluding v) are called the *monotone vertices*. In other words, a monotone vertex of a path has one edge incident from the right and one edge incident from the left.

Lemma 1. *The zigzag path of a pseudo-triangulation T has the following properties:*

1. *It starts at e_s, ends at e_f, and contains no edge twice. Its intersections with l are ordered along l.*
2. (Empty Pseudo-Triangle Property) *The area bounded by the path between two consecutive signposts and the line l is an empty pseudo-triangle, i.e., it contains no points of S in its interior.*
3. *All vertices of the path which are monotone vertices of an empty pseudo-triangle in Property 2 are pointed in T.*

Proof. Property 1 is true by construction. Properties 2 and 3 can be proved inductively by successive elimination of edges e which are not signposts. Each removal will merge two adjacent pseudo-triangles into one. Let e' and e'' be e's neighboring intersecting edges with l. Suppose that (e', e) and (e, e'') lean in the same direction, say, to the left. Let t_1 and t_2 be the pseudo-triangles on the left side of l to which (e', e) and (e, e'') belong, respectively. The left endpoint of e must be a corner (turning point) of t_1 or t_2 (or both), because it cannot be incident to two angles larger than π.

Thus, if we remove e, t_1 and t_2 will merge into a single pseudo-triangle, which is empty. All its monotone vertices were already monotone vertices on the side chains of t_1 or t_2; hence, by induction, they are pointed in T. ◻

Lemma 2. *The zigzag path of a pseudo-triangulation T is the unique chain of edges α in T which satisfies Properties 1–3 of Lemma 1.*

Here, a chain of edges is taken in the graph-theoretic sense, as a walk (or path) in the graph.

Proof. The proof is based on the following easy observation, see Figure 4.

Proposition 2 *Let t be a pseudo-triangle on one side of l, with a segment of l forming a side of t. The other two sides of t are formed by edges of T. Suppose that t contains no points of S in its interior and all monotone vertices of t are pointed in T. Let e' and e'' denote the two edges of T on the boundary of t which intersect l. Then any edge e of T which intersects the interior of t intersects l. Moreover, any two of these edges (including e' and e'') lean in the same direction as e' and e''.*

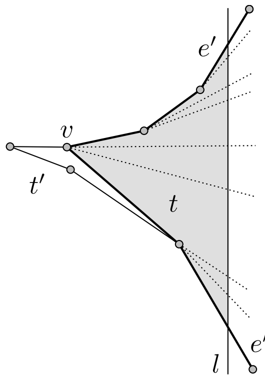

Fig. 4. The pseudo-triangle t in Prop. 2. The dotted lines are some possible locations for the edges e. t' is an alternative pseudo-triangle in the proof of Lemma 2.

Now, to prove the Lemma 2, let us consider two successive intersections e' and e'' of the chain α with l and the pseudo-triangle t formed between them. By Proposition 2, all edges of T intersecting l between e' and e'' lean in the same direction. Hence there can not be a signpost of T between e' and e'' which implies that every signpost is part of the path α.

Let us consider three successive crossings e', e'', e''' between α and l. Between two crossings, α forms a pseudo-triangle with l; hence the two crossing edges lean to the side on which this piece of α lies. Since α crosses from one side of l to the other side at each crossing, the pairs (e', e'') and (e'', e''') lean in different directions. Let $e'' = e_i$ in the ordered set of intersections of T with l. Proposition 2 implies that (e_{i-1}, e_i) leans on the same side as (e', e'') and (e_i, e_{i+1}) leans on the same side as (e'', e'''). Hence e_i is a signpost of T.

At this point we have established that the intersections of α with l are exactly the signposts of T. We still have to check that α bounds a unique pseudo-triangle between two signposts. Let t be the pseudo-triangle between two signposts e' and e'' in the zigzag path $\alpha_l(T)$, and let v be its turning point. Suppose, for the sake of deriving a contradiction, that α bounds a different pseudo-triangle t' between e' and e''. Since t bounds the face in T^* obtained by removing all crossing edges between e' and e'' from T and since α does not contain these edges, we must have $t \subset t'$. Because t' has no interior vertices, it must have all vertices of t on its boundary. If v is the turning point of t', then $t' = t$. So let us assume w.l.o.g. that v lies on the upper chain of t', see Figure 4. Then the lower side chain of t starts with an edge going from v into the interior of t' and ends at e''. This initial edge contradicts Proposition 2 applied to t'. □

The properties of Lemma 1 allow us to define a pseudo-triangulation path of a pointgon without reference to a particular pseudo-triangulation.

Definition 1 (Zigzag Path of a pointgon) *Let (S, P) be a pointgon and let l be a cut segment. A* pseudo-triangulation zigzag path *of (S, P) with respect to l is a non-crossing path in P using vertices of S with the following properties:*

1. *It starts at e_s and ends at e_f. Its intersections with l are ordered along l.*
2. (Empty Pseudo-Triangle Property) *The area bounded by the path between two consecutive intersections with l and the line l is an empty pseudo-triangle.*
3. *The path respects the pointedness property at S, i.e., every vertex of S is pointed in $\alpha \cup P$.*

We denote by $\Pi_l(S,P)$ the set of all paths for a pointgon (S,P) with respect to a line l, i.e.,

$$\Pi_l(S,P) = \{\, \alpha_l(T(S,P)) \mid T \text{ is a pointed pseudo-triangulation of } (S,P)\,\}.$$

Lemma 3. *Let α be a path for (S,P) with respect to the cut segment l.*

1. *$P \cup \alpha$ can be extended to a pointed pseudo-triangulation of (S,P) .*
2. *Let T be any pointed pseudo-triangulation of (S,P) which contains α. Then α is the zigzag path of T with respect to l. The intersections of α with l are the signposts of T.*

5 Making Progress – Trivial Paths

A zigzag path α for a pointgon (S,P) that runs completely along the boundary of P does not cut P into pieces and we will not make any progress by using α. But we will see that the only case where we cannot continue is in fact a single pseudo-triangle without interior points. Then clearly, there is only the "trivial" pseudo-triangulation and we can solve the problem directly.

For a set S of points in the plane a direction d is *feasible* if no line spanned by two points of S is parallel to d. A feasible line is a line with a feasible direction.

Theorem 1. *If δP of a pointgon (S,P) contains at least 4 convex vertices or if (S,P) has at least one inner point, then for each given feasible direction there exists a line l such that all path in $\alpha_l(P)$ are non-trivial.*

Proof. (Sketch) Any trivial path α is a part of δP, i.e., there are no signposts between start and final edge. By Definition 1 two signpost are always connected via exactly one turning point which implies that if the part of δP in consideration contains two convex corners no trivial path can be part of it.

W.l.o.g. let the given orientation of l be vertical. We will use l as a sweep-line for P, moving from left to right. We consider any convex corner of δP, any inner point of (S,P), as well as the left- and rightmost point of any side-chain of δP as an event. There are five different types of events: (1) A corner c of δP, such that after the sweep line passes through c the two incident side chains of δP form a wedge opening to the right. (2) Two of these wedges coalesce at a vertex. (3) A wedge is 'split' by a vertex of δP into two wedges. (4) One of the side chains of a wedge swept by l ends in a convex corner of δP. (5) An inner point of (S,P). A careful case analysis (full details can be found in the journal version) shows that during the sweep there always occurs a position for l such that any path with respect to l and P is non-trivial. □

6 Generating pseudo-triangulation zigzag paths

We will construct the zigzag paths incrementally, edge by edge, starting from the start edge e_s. In each stage, there may be several possibilities to continue the path. All these possibilities are explored in a backtracking tree. The important point of our construction is that one can never get stuck. There is always at least one way to continue. This means that the total work of the algorithm can be bounded in terms of the number of paths generated. This is in sharp contrast to the zigzag path of a triangulation, which cannot be generated in this way without backtracking [2].

Definition 2 (Partial path of a pointgon) *A* partial path α *of a pointgon* (S, P) *with respect to a line* l *is a noncrossing chain starting with* e_s *with the following properties.*

1. *The intersections of* α *with* l *are ordered from top to bottom on* l
2. *The path respects the pointedness property at every vertex of* S*, i.e., every vertex of* $P \cup \alpha_l(S, P)$ *is pointed.*
3. *The area bounded by the path between two consecutive intersections with* l *and the line* l *is an empty pseudo-triangle.*
4. *If we extend the last segment of* α *until it hits* l*, the area bounded by this extension, the line* l*, the path from the last intersection with* l *to the end of* α *is an empty pseudo-triangle. (If the last edge of* α *moves away from* l*, then this last segment is not included in this pseudo-triangle. In particular, if the last edge intersects* l*, the pseudo-triangle degenerates into a line segment and the condition is trivially fulfilled.)*

For a partial path α^* we define the *lower opposite wedge* as follows: we extend the last edge of α^* across l to the opposite side of the current endpoint of α^* until it hits δP. The area in P below this line and on the opposite side of l is the *lower opposite wedge* (the shaded region in Figure 5a).

Lemma 4. *A partial zigzag path* α *can be extended to a complete zigzag path if and only if the lower opposite wedge contains a point of* S*.*

Proof. Suppose that such a point exists. We will construct an extension for α, without destroying the pointedness of any vertex. W.l.o.g., assume that α ends on the right side of l in the point a. α may partition P into several regions. We look at the region R which contains a and the left endpoint b of e_f, see Figure 5.b. The desired extension of α must pass through R. If the angle at a in R is bigger than π, then we walk along the boundary of R away from l to the next point a' where the angle in R is less than π, see Figure 5.a. (This is done to maintain pointedness at a.) If the angle at a in R is smaller than π, we set $a' = a$. Similarly we construct a point b' by starting at b and walking away from l to the first small angle.

Now we take the following path β from a' to b': Start at a', follow the boundary of R to a, follow the extension of the last edge towards l, follow l to its intersection with the lower edge e_f, follow e_f to its left endpoint b, and continue

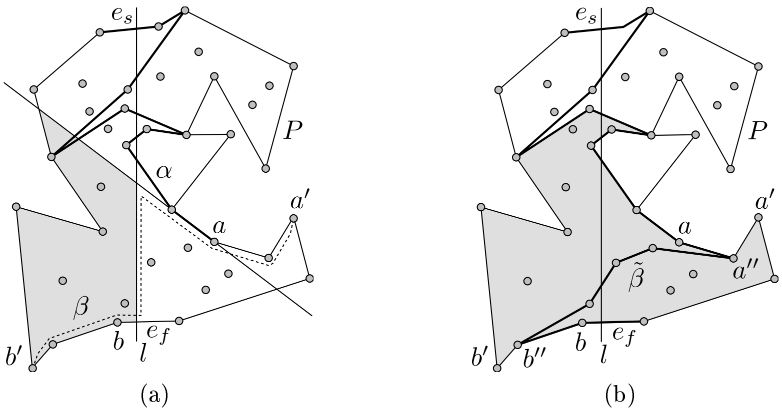

Fig. 5. (a) The lower opposite wedge of a partial zigzag path α and the path β in the proof of Lemma 4. (b) The region R (shaded) and the extension of α.

to b'. The path β runs in P and does not intersect α. Now we take the shortest path $\tilde{\beta}$ homotopic to β. In other words, we consider β as a string and pull it taut, regarding the points of S as obstacles, see Figure 5.b. The path $\tilde{\beta}$ may share some initial part of the boundary of R between a' and a with β, and it will split off at some vertex a''. Similarly we can define such a point b'' towards the end of $\tilde{\beta}$. The path from a to a'', from there to b'' via $\tilde{\beta}$, and from there to b and e_f extends α to a zigzag path. Since the additional edges come from a geodesic path between two convex vertices, pointedness is maintained.

On the other hand, suppose that the lower opposite wedge is empty. Then the extension of the last edge hits the lower edge e_f in an interior point, and the lower opposite wedge is a triangle. Clearly, the path cannot be extended by an edge which leads to a point on the other side of l without violating Property 3 of Definition 2. If α is extended without crossing l, this makes the lower opposite wedge smaller, and hence there is no way to complete the zigzag path. □

Note that the construction in the above proof is only carried out for the purposes of the proof; it is not performed by our algorithm.

Now, if we have a partial path satisfying the condition of Lemma 4, we have to find all edges that may be used to extend the path. We will show that this can be done in $O(n)$ time, after some preprocessing of the point set which takes $O(n^2)$ time and storage. In the preprocessing phase we compute and store the circular order of the edges from each point to all other points of S in $O(n^2)$ time [10]. At this stage, we can already eliminate edges which do not lie inside$P$.

The next edge which is added to a partial path must fulfill Properties 2 (pointedness) and 3 (empty area) of Definition 2, the non-empty opposite wedge condition of Lemma 4, and it must not cross the previous edges of the path.

Let a be the endpoint of α^* and assume w.l.o.g. that it lies on the right side of l. Take a line through the last edge of α^* and rotate it counterclockwise around

a until it hits the first point b on the right side of l. All points that are hit by this line and that are visible from a (including b) are candidates for the next point that satisfy the empty area condition, see Figure 6. If the last edge has moved away from l, then this holds for points on both sides of l. Otherwise, the next point must either be b or on the opposite side of l.

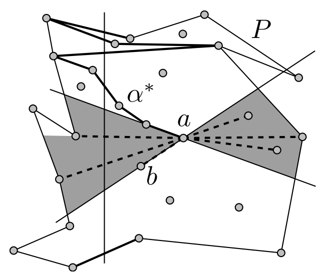

Fig. 6. The possible continuations of a partial path.

This set of continuation points depends only on a and on the last edge of α^*, and hence it can be determined in the preprocessing phase. Similarly the condition of Lemma 4 can be checked beforehand and edges which violate the condition are eliminated. The only conditions which have to be checked dynamically are the pointedness and non-crossing conditions.

Pointedness is easy to maintain: For each vertex a of S we store the unique angle between two incident edges which is larger than α. If a new edge incident to a is inserted, we see whether it falls into the wedge of the big angle, and if so, we either updated the big angle or we reject the edge because it destroys pointedness, in constant time. During the generation of all paths in the enumeration tree, edges are removed in the reverse order as they were inserted, so it is easy to maintain the big angle in stack-like manner.

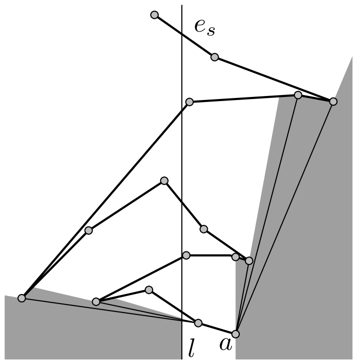

Fig. 7. The visibility polygon.

Now we still have to check that the new edge does not cross the partial path α^*. We show that we can do this, for all possible continuation edges from the endpoint a, in linear time.

We can easily check whether any edge intersects α^* if we know the *visibility polygon* from a with respect to α^*, see Figure 7. The visibility polygon is stored as a sequence of consecutive angular intervals together with an identification which edge of α^* is first first hit by a ray from a in that interval. We will show below in Lemma 7 how to exploit the special structure of the path to compute the desired visibility polygon in $O(n)$ time in an easy way.

Lemma 5. *For a given partial path all possible edges which extend it to a legal partial path satisfying the condition of Lemma 4 can be found in $O(n)$ time.*

Proof. For the last edge of the partial path leading to the endpoint a, we have already precomputed the set of possible extension edges for which the following conditions are maintained: the empty pseudo-triangle condition (Property 3 of Definition 2), the non-empty opposite wedge condition of Lemma 4, and the edge lies inside P. This list of $O(n)$ candidate edges is given in cyclic order. We compute the visibility polygon of a with respect to α^* in $O(n)$ time, by Lemma 7, and we merge the candidate edges into the cyclic order of the visibility polygon, checking for each edge whether it intersects α^* in constant time.

As mentioned above, pointedness can also be checked in constant time for each edge. □

We will now sketch how to construct the (relevant part of) the visibility polygon in an easy way. Suppose that the current endpoint a is on the right of l and let us concentrate on the possible continuation edges to the right of a (moving further away from l). In this case we are only interested in the part of the visibility polygon that lies to the right of a.

Lemma 6. *Suppose a is on the right side of l and let r be a ray which emanates from a to the right (away from l). Let e_i be the first edge of α^* which is hit by r. Then all other edges of α^* which are hit by r come before e_i on α^*.*

Proof. (Sketch.) This is based on the fact that each of the pseudo-triangles formed by l and the parts of α^* right of l consist of two x-monotone chains from l to the right, meeting at a corner vertex, and that the intersections of α^* with l occur in the correct order (Property 1 of Definition 2). □

It follows that we can simply compute the right part of the visibility polygon by scanning the edges of α^* in reverse order, starting at a. The edges which are scanned so far will cover some angular region Q around a starting at the vertical upward direction. This part of the visibility polygon is already a correct part of the final visibility polygon. We only have to wait until some edge of α^* appears behind the already seen edges at the right edge of Q, and extend Q accordingly.

The same arguments apply to possible continuation edges to the left of a. Such an edge can only cross α^* if it crosses l. For the part of the visibility polygon that lies to the left of l, the above arguments can be applied. Thus we have:

Lemma 7. *The part of the visibility polygon of a with respect to α^* which lies to the right of a or to the left of l can be computed in $O(n)$ time.*

We can now enumerate all zigzag paths by scanning the enumeration tree. Note that the path is not automatically complete when it reaches an endpoint of the final edge e_f, but only when the edge e_f itself is inserted into the path. (Lemma 4 also holds when the partial path ends at an endpoint of e_f. In this case the continuation is always guaranteed.)

Theorem 2. *For a pointgon (S, P) and a line l we can enumerate the set $\alpha_l(S, P)$ of pseudo-triangulation zigzag paths in time $O(n^2 + n^2|\alpha_l(S, P)|)$ and space $O(n^2)$.*

Of course, this space bound does not include the space which is necessary to store all paths.

Proof. The enumeration tree has $|\alpha_l(S, P)|$ leaves. Since a zigzag path has length $O(n)$, being a noncrossing set of edges, the enumeration tree has depth $O(n)$, and hence $O(n|\alpha_l(S,P)|)$ nodes. By Lemma 5, we spend $O(n)$ per node. The $O(n^2)$ preprocessing time was already mentioned. The $O(n^2)$ space includes the space for storing all cyclic orders and the stack of large angles for each point. $\square$

Note that the time bound is overly pessimistic. In practice, the tree can be expected to be "bushy" and have only $O(|\alpha_l(S,P)|)$ nodes.

References

1. P. K. Agarwal, J. Basch, L. J. Guibas, J. Hershberger, and L. Zhang. Deformable free space tilings for kinetic collision detection. In B. R. Donald, K. Lynch, and D. Rus (eds.), *Algorithmic and Computational Robotics: New Directions (Proc. 5th Workshop Algorithmic Found. Robotics)*, pages 83–96. A. K. Peters, 2001.
2. O. Aichholzer. The Path of a Triangulation. In *Proc. 15th ACM Symp. Computational Geometry*, pages 14–23, 1999.
3. D. Avis. lrslib Software: Reverse search algorithm for vertex enumeration/convex hull problems. `http://cgm.cs.mcgill.ca/~avis/C/lrs.html`
4. D. Avis and K. Fukuda. Reverse search for enumeration. *Discrete Appl. Math.*, 65:21–46, 1996.
5. S. Bespamyatnikh. Enumerating Pseudo-Triangulations in the Plane. In *Proc. 14th Canad. Conf. Comp. Geom.*, pages 162–166, 2002.
6. S. Bespamyatnikh. An efficient algorithm for enumeration of triangulations. *Comp. Geom., Theory Appl.*, 23(3):271–279, 2002.
7. H. Brönnimann, L. Kettner, M. Pocchiola, and J. Snoeyink. Counting and enumerating pseudo-triangulations with the greedy flip algorithm. Manuscript, 2001.
8. Bernard Chazelle, Herbert Edelsbrunner, Michelangelo Grigni, Leonidas J. Guibas, John Hershberger, Micha Sharir, and Jack Snoeyink. Ray shooting in polygons using geodesic triangulations. *Algorithmica*, 12:54–68, 1994.
9. A. Dumitrescu, B. Gärtner, S. Pedroni, and E. Welzl. Enumerating triangulation paths. *Comp. Geom., Theory Appl.*, 20:3–12, 2001.
10. H. Edelsbrunner and J. O'Rourke and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. In *SIAM J. Comput.*, 15:341–363, 1986.
11. K. Fukuda. Software: cdd and cddplus. `http://www.cs.mcgill.ca/~fukuda/soft/cdd_home/cdd.html`
12. Michael Goodrich and Roberto Tamassia, Dynamic ray shooting and shortest paths in planar subdivisions via balanced geodesic triangulations. *J. Algorithms* 23:51–73, 1997.
13. D. Kirkpatrick, J. Snoeyink, and B. Speckmann. Kinetic collision detection for simple polygons. *Intern. Journal Comp. Geom. Appl.*, 12(1 & 2):3–27, 2002.
14. David Orden, Francisco Santos The polytope of non-crossing graphs on a planar point set. Manuscript, February 2003, arXiv:math.CO/0302126.
15. M. Pocchiola and G. Vegter. Topologically sweeping visibility complexes via pseudo-triangulations. *Discrete Comp. Geom.*, 16:419–453, 1996.
16. G. Rote, F. Santos, and I. Streinu. Expansive motions and the polytope of pointed pseudo-triangulations. Manuscript, FU-Berlin, September 2001.
17. B. Speckmann and C. Tóth. Allocating Vertex π-guards in Simple Polygons via Pseudo-Triangulations. In *Proc. 14th Symp. on Discr. Algor.*, pages 109–118, 2003.
18. I. Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. In *Proc. 41st FOCS*, pages 443–453, 2000.