Pointed Binary Encompassing Trees: Simple and Optimal*

Michael Hoffmann[1] Bettina Speckmann[2] Csaba D. Tóth[3]

Abstract

For n disjoint line segments in the plane we construct in optimal $O(n \log n)$ time and linear space an encompassing tree of maximum degree three such that at every vertex all incident edges lie in a halfplane defined by the incident input segment. In particular, this tree is *pointed* since every vertex has an incident angle greater than π. Such a pointed binary tree can be augmented to a minimum pseudo-triangulation. It follows that every set of disjoint line segments in the plane has a constrained minimum pseudo-triangulation whose maximum vertex degree is bounded by a constant.

1 Introduction

Spanning trees defined on disjoint objects in the plane are fundamental structures in computational geometry. Complex planar objects are often modeled by their boundary polygons which, in turn, can be represented as planar straight line graphs (PSLGs). An *encompassing graph* for a PSLG G is a *connected* PSLG on the same vertex set that contains all edges of G, see Figure 1(b) for an example. Constrained Delaunay triangulations [15] are well-known examples of encompassing graphs.

Particularly well-studied are encompassing trees for disjoint line segments in the plane. In this context, a set of disjoint segments is regarded as a PSLG that is a perfect matching. It is easy to construct an encompassing tree for any given PSLG in $O(n \log n)$ time; a triangulation, for instance, of the free space around n disjoint line segments is an encompassing graph. Therefore, most previous research—as well as this paper—focuses on optimizing various parameters (length, degree, etc.) of encompassing trees.

Encompassing trees for segments. Let $\mathrm{ed}(n)$, $n \in \mathbb{N}$, denote the minimum integer such that any set of n disjoint line segments in the plane admits an encompassing tree of maximum vertex degree $\mathrm{ed}(n)$. A simple construction (Figure 1(a)) shows that not every set of n disjoint segments in the plane admits an *encompassing path*, and so $\mathrm{ed}(n) \geq 3$, for $n \geq 6$.

Somewhat surprisingly, it turns out that $\mathrm{ed}(n)$ does not depend on n. Bose, Houle, and Toussaint [4] showed that $\mathrm{ed}(n) \leq 3$, for all $n \in \mathbb{N}$, and hence $\mathrm{ed}(n) = 3$, for all $n \geq 6$. They also gave an $O(n \log n)$ time algorithm for constructing such an encompassing tree. Both the degree bound and the runtime are best possible (the latter in the algebraic computation tree model).

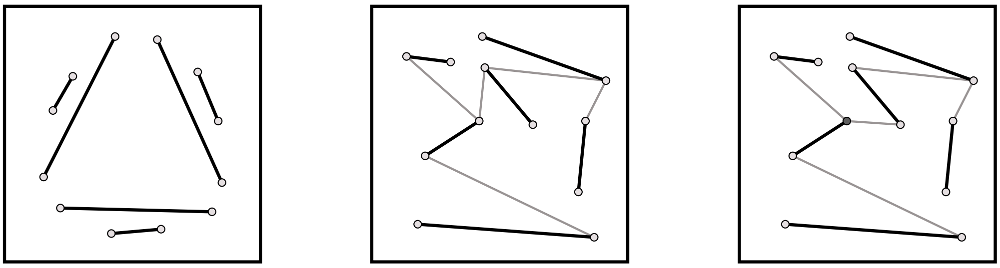
The situation is slightly different for *minimum encompassing trees* where edges are weighted according to the Euclidean distance between their endpoints. Bose and Toussaint [5] discovered families of disjoint line segments for which every minimum encompassing tree requires a vertex of degree seven. Conversely, a minimum weight encompassing tree of maximum degree seven can always be obtained greedily.

*Preliminary versions of this work were presented at the 14th Annual Fall Workshop on Computational Geometry (Cambridge, MA, 2004), the 21st European Workshop on Computational Geometry (Eindhoven, 2005), and the 21st Annual Symposium on Computational Geometry (Pisa, 2005).

[1]Institute of Theoretical Computer Science, ETH Zürich, hoffmann@inf.ethz.ch

[2]Department of Mathematics and Computer Science, TU Eindhoven, speckman@win.tue.nl

[3]Department of Mathematics, MIT, Cambridge, MA, toth@math.mit.edu



(a) (b) (c)

Figure 1: Six segments which do not admit an encompassing path (a), and two encompassing trees: all vertices are pointed (b), the dark vertex is not pointed (c).

Hoffmann and Tóth [11] showed that for disjoint segments that are not all collinear there is always a *Hamiltonian* encompassing graph of maximum degree three. On the other hand, if segments are allowed to share endpoints then it is NP-complete to decide whether or not a given set of segments admits an encompassing (Hamiltonian) circuit [18]. For the special case of convexly independent segments (for no segment both endpoints are interior to the convex hull) Rappaport et al. [19] gave a polynomial time algorithm.

In this paper, we extend the result of Bose, Houle, and Toussaint [4]. For n disjoint line segments in the plane we construct a binary encompassing tree with an additional useful property: *pointedness*.

Pointed PSLGs. In recent years, a relaxation of triangulations, called *pseudo-triangulations*, has received considerable attention. The faces of a pseudo-triangulation are bounded by three concave chains, rather than by three line segments. More formally, a pseudo-triangle is a planar polygon that has exactly three convex vertices with internal angles less than π, all other vertices are concave. Pseudo-triangulations, also called *geodesic triangulations*, were originally studied for convex sets and for simple polygons in the context of visibility [16, 17] and ray shooting [6, 8]. In the last few years they also found application in robot motion planning [22], kinetic collision detection [1, 14], and guarding [21].

Of particular interest are the so-called *minimum pseudo-triangulations*, which have the minimum number of pseudo-triangular faces among all possible pseudo-triangulations of a given domain. They were introduced by Streinu [22], who proved that every minimum pseudo-triangulation of a set S of n points consists of exactly $n-2$ pseudo-triangles. Minimum pseudo-triangulations are also referred to as *pointed pseudo-triangulations* since every vertex v of a minimum pseudo-triangulation is pointed, that is, v has an incident region whose angle at v is greater than π. For example, the dark vertex in Figure 1(c) is not pointed. The converse is also true: If every vertex of a pseudo-triangulation is *pointed* (it has an incident angle greater than π) then this pseudo-triangulation has exactly $n-2$ pseudo-triangles and is therefore minimum.

Pseudo-triangulations, just like triangulations, are PSLGs. But while triangulations of a planar point set can have arbitrarily high vertex degree, there is always a pseudo-triangulation of vertex degree at most five [13]. Bounded vertex degree is a useful property for many applications, since it enables local operations or updates in constant time.

Streinu [22] showed that every pointed PSLG—that is, a PSLG where each vertex is pointed—can be completed to a pointed pseudo-triangulation by greedily adding edges while maintaining pointedness. But this approach does not provide any guarantee regarding the vertex degree. On the other hand, pointed spanning trees are not omnipresent in planar structures: Aichholzer, Huemer, and Krasser [3] presented triangulations which do not contain any pointed spanning tree. Furthermore, both the algorithm of Bose, Houle, and Toussaint [4] and that of Hoffmann and Tóth [11] violate pointedness due to their proof techniques. A natural question is therefore: Given a set of disjoint line segments, can one construct an encompassing tree that maintains the degree bound of previous algorithms but also ensures that the resulting spanning tree is pointed?

Results. We present an algorithm for constructing an encompassing tree that respects pointedness *and* has maximum vertex degree three.

Theorem 1 *For any set of disjoint line segments in the plane, there is an encompassing tree of maximum degree three such that for every vertex v all incident edges lie in a halfplane bounded by the line through the segment of S which v is an endpoint of. For n line segments, such an encompassing tree can be computed in $O(n \log n)$ time and $O(n)$ space.*

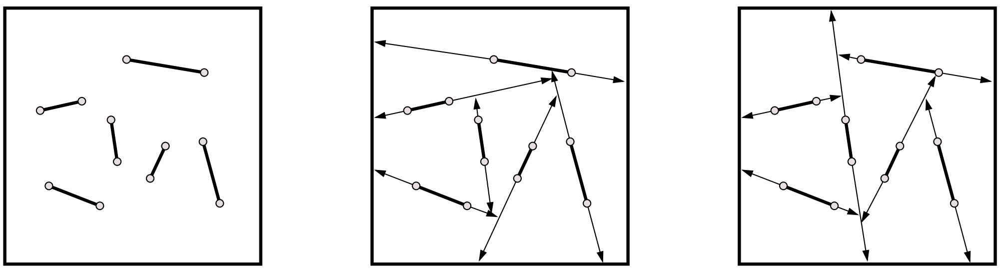
Recently, Aichholzer et al. [2] showed that a bounded degree pseudo-triangulation constrained to contain a Hamiltonian circuit (a simple polygon) always exists, with a degree bound of seven. With the help of a pointed binary encompassing tree we can extend these results to pseudo-triangulations constrained to contain disjoint line segments.

Theorem 2 *For any set of disjoint line segments in the plane, there is an encompassing pointed pseudo-triangulation with maximum vertex degree ten.*

Organization. In Section 2 we prove Theorem 1 via a *tunnel graph* that is defined for a convex partition of the free space around the n input segments. The construction of *connected* tunnel graphs is presented in Section 3 using a dynamic data structure to test connectivity in plane graphs [7]. In Section 4 we present a simple geometric algorithm for our problem that does not rely on this generic tool. Finally, Section 5 shows how to combine the pointed binary encompassing tree with the algorithm of Aichholzer et al. [2] to construct a pointed pseudo-triangulation of maximum vertex degree ten. We conclude with a few open questions in Section 6.

2 Tunnel Graphs

Convex partition and cells. The free space around n disjoint line segments in the plane can be partitioned into $n+1$ convex cells by the following well known partitioning algorithm. (For simplicity, we assume that no three segment endpoints are collinear.) For every segment endpoint p of every input segment s_p, extend s_p beyond p until it hits another input segment, a previously drawn extension, or to infinity. There may be many different partitions depending of the order in which we consider the segment endpoints, but the number of convex cells is always $n+1$ (see Fig. 2).



(a) (b) (c)

Figure 2: Disjoint segments (a), and two of their convex partitions (b) and (c).

Tunnel graphs. Consider a set of disjoint segments S in the plane and a convex partition $P(S)$ obtained by the above algorithm. Let us assign every segment endpoint p to an incident cell $\tau(p)$ of the partition. We define the *tunnel graph* $T(S, P(S), \tau)$ for S, a partition $P(S)$, and an assignment τ as follows: The nodes of T correspond to the convex cells of $P(S)$. Two nodes a and b are connected by an edge if and only if there is a segment $pq \in S$ such that $\tau(p) = a$ and $\tau(q) = b$. The tunnel graph is clearly planar; as it has $n+1$ nodes and n edges, it is connected if and only if it is a tree.

Theorem 3 *For any set S of n disjoint line segments, one can construct in $O(n \log n)$ time and linear space a convex partition $P(S)$ and an assignment τ such that the tunnel graph $T(S, P(S), \tau)$ is a tree.*

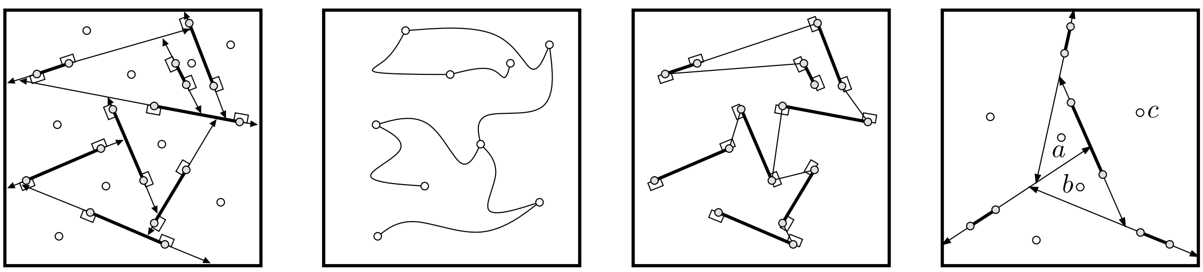
Note that the choice of the convex partition is important in Theorem 3: Figure 3(d) shows four disjoint line segments and a convex partition such that there is no assignment for which the tunnel graph is connected. (Consider the endpoints of the segment s: The left endpoint is the only segment incident to cell a and must hence be assigned to a. Similarly, the right endpoint of s has to be assigned to cell b. But then regardless of the assignment for the other points, $\{a, b\}$ is always a component of size two in the tunnel graph.) We obtain Theorem 1 as a corollary of Theorem 3.

Proof. [of Theorem 1] Consider a partition $P(S)$ and an assignment τ provided by Theorem 3. We construct a binary encompassing tree as follows: In each cell connect all segment endpoints assigned to it by a simple path; for example, connect them in the order in which they appear along the boundary of the cell.

The resulting graph is clearly a PSLG that contains all the input segments. The maximum degree is three because we add at most two new edges at every segment endpoint. It remains to prove connectivity. Let p and r be two segment endpoints. We know that the tunnel graph is connected, so there is an alternating sequence of cells and segments

$$(a_1{=}\tau(p), p_1q_1, a_2, \ldots, p_{k-1}q_{k-1}, a_k{=}\tau(r))$$

such that $\tau(p_i) = a_i$ and $\tau(q_i) = a_{i+1}$, for every $1 \leq i < k$. As all segment endpoints assigned to the same cell are connected, this path corresponds to a path in the constructed graph. □



(a) Partition. (b) Tunnel Graph. (c) Encompassing Tree. (d) Disconnected.

Figure 3: An example for a partition with an assignment (a), the corresponding tunnel graph (b), the resulting tree (c). A partition for which no assignment gives a connected tunnel graph (d).

3 Constructing a Connected Tunnel Graph

This section presents an algorithmic proof of Theorem 3. Consider a set S of n disjoint line segments in the plane and let R be an axis-parallel box which contains all segments of S in its interior. We use a two-phase algorithm to compute

- a partition $P(S)$ of the free space around the segments into $n+1$ convex cells; and
- an assignment of the segment endpoints to incident cells

The first phase is a left-to-right sweep: We extend every input segment beyond its right endpoint until the extension hits another segment, another extension, or the boundary of R. If two extensions meet, an arbitrary one continues and the other one ends (Figure 4(b)). The free space of the input segments and their right extensions is a simply connected set $C_0 \subset R$. In the second phase, the left extensions of the segments are inserted one-by-one in an arbitrary order.

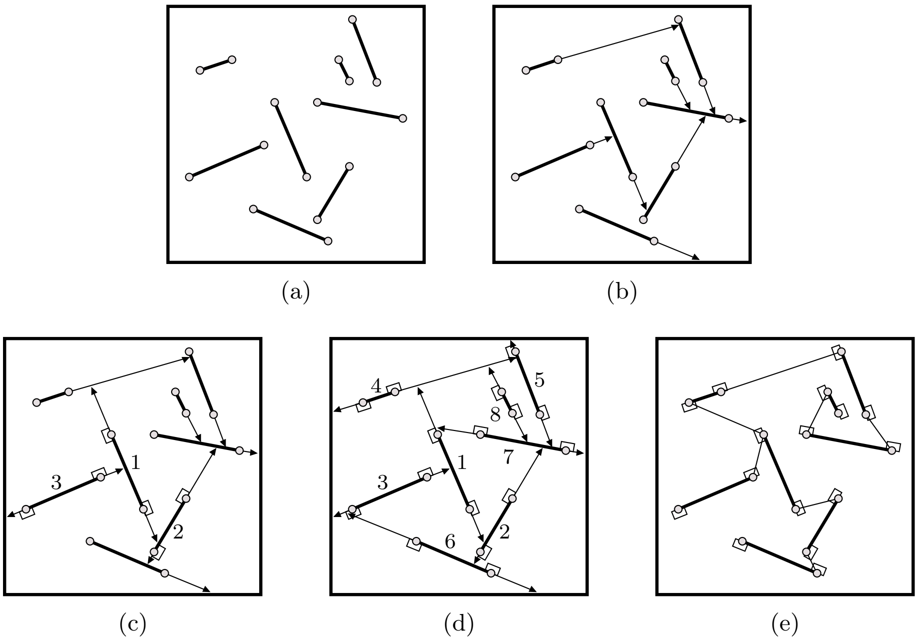


Figure 4: Constructing the partition: First all right extensions (b), then the left extensions are inserted one by one (c) and (d), and from the final partition together with the assignment we can construct the encompassing tree (e).

Denote by $\mathcal{A}_i$, $0 \le i \le n$, the arrangement of the input segments, all their right extensions, and the left extensions of $s_1, \ldots, s_i$. At the beginning of the second phase, no left extension has been drawn yet. We face the arrangement $\mathcal{A}_0$ in which there is only one single cell C_0. After the second phase, the arrangement to be considered is $\mathcal{A}_n$, which consists of $n+1$ convex cells.

We define the assignment τ on the endpoints of s_i, $i = 1, 2, \ldots, n$, as soon as the left extension γ_i of s_i is inserted. At this point we have an arrangement $\mathcal{A}_{i-1}$ that consists of i cells and a partial assignment τ on the endpoints of the first $i-1$ segments. $\mathcal{A}_{i-1}$ and τ define a tunnel graph T_{i-1} on i nodes. We choose the assignment at the endpoints of s_i inductively such that the resulting tunnel graph T_i remains connected. Clearly T_0 is connected because it is a graph on one node only.

For the induction step consider the ray γ_i that splits a cell C_i of $\mathcal{A}_{i-1}$ into two cells C'_i and C''_i of $\mathcal{A}_i$. Correspondingly, a node of T_{i-1} is split into two nodes that are in different components of the resulting graph T'_{i-1}. The left endpoint p_i of s_i is incident to both C'_i and C''_i because

p_i is the source of the ray γ_i that separates both cells. The right endpoint q_i, however, may be incident to neither C_i' nor C_i''. We always assign q_i to the cell lying above q_i. Then p_i is assigned to C_i' or C_i'', whichever lies in the component of T_{i-1}' which does not contain $\tau(q_i)$. As T_{i-1}' has exactly two components, this assignment ensures that the resulting tunnel graph T_i is connected

It remains to show the time and space bounds claimed in Theorem 3. The arrangement $\mathcal{A}_n$ can be constructed using two standard line sweep algorithms in $O(n \log n)$ time and linear space. First, the right extensions are handled in a left-to-right sweep. Then, all left extensions are pre-computed in a single right-to-left sweep; whenever two left extensions meet, the one with the smaller index according to the insertion order continues and the other ends. The combinatorial complexity of $\mathcal{A}_n$ is $O(n)$ and we can store it in a standard DCEL structure. Any type of incidence and adjacency information—such as which two cells a segment endpoint is adjacent to—can be extracted from this arrangement.

In the second phase, the pre-computed left extensions are inserted one-by-one. To decide on the assignment for the left segment endpoints, we need to maintain the connectivity information about the tunnel graph under construction. For any arbitrary order of the extensions and any assignment for the segment endpoints, the graph formed by the tunnel edges is plane (that is, a planar graph with a given embedding). Hence, we can employ a dynamic data structure of Eppstein et al. [7] which maintains the connected components of a plane graph under edge insertions and vertex splits in $O(\log n)$ time per update and $O(n)$ space. This structure can answer connectivity queries for any two vertices in $O(\log n)$ time.

Therefore, the algorithm can be implemented in $O(n \log n)$ time and $O(n)$ space.

4 A Simple Geometric Algorithm

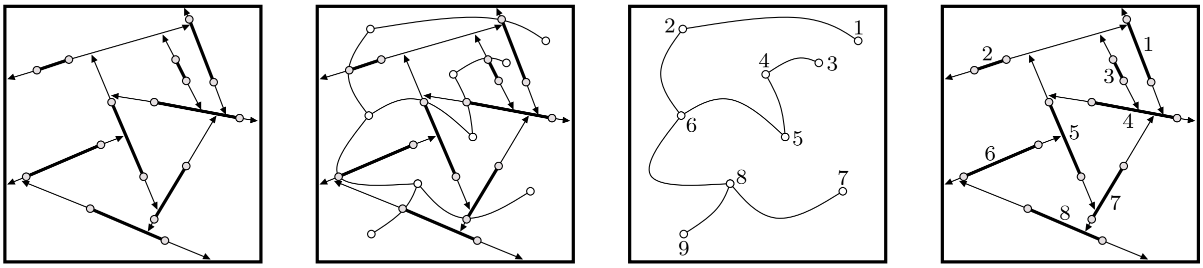
In the previous section, we proved Theorem 3. In this section we present an alternative, simple, and geometric algorithm that computes a tunnel graph in $O(n \log n)$ time and $O(n)$ space without relying on the dynamic data structure by Eppstein et al. [7].

We first compute the arrangement $\mathcal{A}_n$ for an arbitrary insertion order with two line sweeps as described above. For every segment $s_i \in S$, we define a left extension curve β_i as follows: β_i is a directed polygonal curve in the arrangement $\mathcal{A}_n$; it starts from the left endpoint p_i along the left extension of s_i and ends when it hits another segment, a right extension, or the bounding box; if it hits another left extension in $\mathcal{A}_n$ then it continues along it in right-to-left direction.

Let us denote by $\mathcal{B}_i$, $0 \leq i \leq n$, the arrangement formed by the input segments, their right extensions, and the first i left extension curves, $\beta_1, \beta_2, \ldots, \beta_i$. The arrangement $\mathcal{B}_0 = \mathcal{A}_0$ has a single (non-convex) face. When all n left extension curves are inserted, we obtain the arrangement $\mathcal{B}_n = \mathcal{A}_n$.

Observe that the proof of Theorem 3 holds true verbatim with curves β_i instead of the rays γ_i. Indeed, we have used only the properties that each β_i splits a cell C_i of $\mathcal{B}_{i-1}$ into two cells C_i' and C_i'' of $\mathcal{B}_i$, and the left endpoint p_i is incident to both C_i' and C_i''.

A particular order of the segments. For computing an assignment τ on the n left segment endpoints in $O(\log n)$ time each, we insert the curves β_i into the arrangement $\mathcal{B}_0$ in a particular order σ. In an arrangement $\mathcal{B}_i$, let us call the face incident to the lower-left corner of the bounding box the *LL-cell*, and all other faces the *terminal cells*. We choose σ such that in every arrangement $\mathcal{B}_i$, $i = 0, 1, \ldots, n$, every terminal cell is a cell of the final arrangement $\mathcal{B}_n = \mathcal{A}_n$. This is equivalent to the property that for every $i = 0, 1, \ldots, n-1$,, the curve β_{i+1} partitions the LL-cell of the arrangement $\mathcal{B}_i$.



(a) (b) (c) (d)

Figure 5: A convex partition (a), every left segment endpoint defines an edge of the auxiliary graph G (b), an order $\hat{\sigma}$ (c), and the corresponding order σ (d).

Computing an order. A suitable order σ of the n input segments can be computed in $O(n)$ time from the arrangement $\mathcal{A}_n$. We define and compute an auxiliary graph G with the help of $\mathcal{A}_n$: The nodes of G correspond to the faces of the arrangement $\mathcal{A}_n$; two nodes of G are connected by an edge if and only if the corresponding cells are incident to a left segment endpoint (see Fig. 5).

Proposition 4 *The auxiliary graph G is connected.*

Proof. Suppose, to the contrary, that G is disconnected. One connected component of G corresponds to a region $Q \subset R$ which is the union of cells of $\mathcal{A}_n$ and the boundary of Q contains no left segment endpoint. That is, there is a cycle in $\mathcal{A}_n$ that is nontrivial (different from the boundary of R) and does not pass through any left segment endpoint. This is impossible because the input segments and their right extensions (i.e., the arrangement $\mathcal{A}_0$) do not form any nontrivial cycle and, similarly, the right extensions do not form any cycle either. Any nontrivial cycle in $\mathcal{A}_n$ must contain a left segment endpoint. We conclude that G is connected. □

Graph G has $n+1$ nodes and n edges, and it is connected, therefore it is a tree. Let (G, r) be a rooted tree where $r \in G$ is the node corresponding to the cell incident to the lower left corner of R. We choose an auxiliary order $\hat{\sigma}$ on the nodes of G: We recursively pick a leaf of the rooted tree (G, r), we insert it into $\hat{\sigma}$ and delete it until we have processed the root r.

We compute the order σ of input segments based on the auxiliary order $\hat{\sigma}$ of cells of $\mathcal{A}_n$. For any $i = 1, 2, \ldots, n$, let the i-th element of σ be the segment whose left endpoint corresponds to the edge of G between the i-th node in $\hat{\sigma}$ and its parent node.

The order σ has the required property: We argue that the first i cells (in $\hat{\sigma}$-order) of the arrangement $\mathcal{A}_n$ are terminal cells in the arrangement $\mathcal{B}_i$ (which is obtained by inserting the curves $\beta_1, \beta_2, \ldots, \beta_i$ in σ-order). We proceed by induction on i. The claim is obvious for $i = 0$, when $\mathcal{B}_0$ has no terminal cells. If it is true for every $i' < i$, then consider $\mathcal{B}_{i-1}$ and insert the left extension curve β_i. Let D_i denote the i-th cell of $\mathcal{A}_n$. The starting point p_i of curve β_i lies on the boundary of D_i. By the definition of $\hat{\sigma}$, all other β-curves whose starting points lie along D_i have already been inserted. Therefore β_i splits the LL-cell of $\mathcal{B}_{i-1}$ into D_i and the LL-cell of $\mathcal{B}_i$.

Union-find forest. We insert the curves β_i, $i = 1, 2, \ldots, n$, one-by-one in σ-order and choose the assignment $\tau(p_i)$ similarly to the proof of Theorem 3. We obtain the arrangement $\mathcal{B}_i$ by inserting the left extension curve β_i into $\mathcal{B}_{i-1}$. Curve β_i splits the LL-cell of $\mathcal{B}_{i-1}$ into the LL-cell of $\mathcal{B}_i$ and a new terminal cell D_i (which is a cell of $\mathcal{A}_n$ by choice of σ). The left endpoint p_i lies on the common boundary of these two new cells. Curve β_i also splits the corresponding node of the tunnel graph T_{i-1} into two nodes, and splits T_{i-1} into a forest T'_{i-1} of two components. To find the correct assignment $\tau(p_i)$, it is enough to decide whether the cell of $\mathcal{B}_i$ above the right endpoint q_i is connected to D_i in T'_{i-1} or not.

For every $i = 0, 1, \ldots, n$, we maintain a data structure for the terminal nodes of $\mathcal{B}_i$. We represent the tunnel edges *between terminal cells* of $\mathcal{B}_i$ in a union-find search forest U_i. (Note that U_i does not represent tunnel-edges between the LL-cell and terminal cells, and so U_i may be disconnected.) We construct U_i from U_{i-1} as follows: We create a new node d_i representing the new terminal cell D_i. By traversing the boundary of D_i, we list all terminal cells connected to D_i by a tunnel edge of T_{i-1}. We merge the corresponding trees of U_{i-1} and the singleton d_i into one tree according to union-by-height. The search forest U_i has height $O(\log n)$.

Finally, we look up the cell above the right endpoint q_i and the terminal cell D_i in the search forest U_i. If they are in different trees, then we assign p_i to D_i, otherwise we assign p_i to the LL-face of $\mathcal{B}_i$. The argument in the proof of Theorem 3 guarantees that we obtain an assignment τ for which the tunnel graph $T(S, P(S), \tau)$ is connected.

Runtime analysis. The arrangement $\mathcal{A}_n$ can be constructed using two line sweep algorithms in $O(n \log n)$ time and $O(n)$ space. The order σ can be computed from $\mathcal{A}_n$ in $O(n)$ time. The curves β_i may overlap, but we compute them only until they split a cell of $\mathcal{B}_i$ into two cells, and $\mathcal{B}_n$ has the same complexity as $\mathcal{A}_n$, that is $O(n)$. Finding a terminal cell in the union-find forest U_i takes $O(\log n)$ time which implies that we can choose the assignment for each left segment endpoint in $O(\log n)$ time. Therefore our algorithm runs in $O(n \log n)$ time and $O(n)$ space.

5 Bounded Degree Pseudo-Triangulations for Disjoint Segments

For the proof of Theorem 2, we combine Theorem 1 with the algorithm of Aichholzer et al. [2], according to which a simple polygon can be pseudo-triangulated such that the degree of every convex vertex is at most four and the degree of every reflex vertex is at most five, that is, every convex (reflex) vertex has at most two (three) new incident edges in addition to the two incident polygon edges. In fact, we use a special case of their result:

Theorem 5 ([2]) *For any weakly simple polygon P with two consecutive convex vertices u and v there is a pointed pseudo-triangulation of the interior of P such that*

- *every reflex vertex of P receives at most three additional edges;*
- *every convex vertex of P receives at most two additional edges;*
- *u receives at most one additional edge;*
- *v does not receive any additional edge.*

Let $\mathcal{D}(T, S)$ denote the arrangement formed by the binary pointed encompassing tree T provided by Theorem 1 and the convex hull $\mathcal{CH}(S)$. The faces of the arrangement $\mathcal{D}(T, S)$ are *weakly* simple polygons.

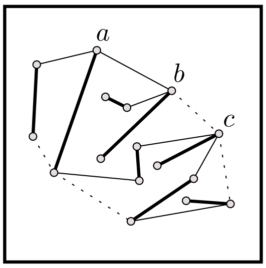


Figure 6: $\mathcal{D}(T, S)$.

Proposition 6 *Every face P of the arrangement $\mathcal{D}(T, S)$ has exactly one edge which is part the convex hull $\mathcal{CH}(S)$ but does not belong to T.*

Proof. If P has only edges that belong to T then T has a cycle. So P has at least one edge from $\mathcal{CH}(S) \setminus T$. But P can not have more than one edge from $\mathcal{CH}(S) \setminus T$, otherwise T would be disconnected. □

We obtain a bounded degree pseudo-triangulation of the input segments by pseudo-triangulating each face of $\mathcal{D}(T,S)$ using the algorithm by Aichholzer et al. [2].

Proof. [of Theorem 2.] Consider a weakly simple face P of the arrangement $\mathcal{D}(T,S)$. By Proposition 6 P has an edge e which is part of $\mathcal{CH}(S)$ but does not belong to T. We label the vertices of e by u and v such that $\overrightarrow{uv}$ is oriented counterclockwise along $\mathcal{CH}(S)$. Now we apply Theorem 5 to P, u, and v.

$\mathcal{D}(T,S)$ contains two types of vertices: Convex hull vertices p_{hull} and interior vertices p_{int}. Every interior vertex p_{int} is pointed and has maximal degree 3. So p_{int} is incident to at most 3 angular domains, one of which is reflex. By Theorem 5, the degree of p_{int} increases by at most $3+2+2=6$ to at most 10.

Every convex hull vertex p_{hull} has degree at most 3 in T and therefore degree at most 5 in $\mathcal{D}(T,S)$. We distinguish three cases according to the type of the two convex hull edges, e_1 and e_2, incident to p_{hull} (see Fig. 6).

(a) If both e_1 and e_2 are edges of T, then p_{hull} is incident to at most 2 angular domains, all of which are convex. In this case, the degree of p_{hull} is at most 3 in $\mathcal{D}(T,S)$, and it increases by at most $2+2=4$ to at most 7.

(b) If either $e_1 \notin T$ or $e_2 \notin T$ (but not both), then p_{hull} is incident to at most 3 angular domains, all of which are convex. In this case, the degree of p_{hull} is at most 4 in $\mathcal{D}(T,S)$. W.l.o.g. we assume $e_1 \notin T$. Then p_{hull} is labeled either u or v with respect to the face of $\mathcal{D}(T,S)$ adjacent to e_1. Hence Theorem 5 implies that the degree of p_{hull} increases by at most $2+2+1=5$ to at most 9.

(c) If $e_1 \notin T$ and $e_2 \notin T$, then p_{hull} is incident to at most 4 angular domains, all of which are convex. In this case, the degree of p_{hull} is 5 in $\mathcal{D}(T,S)$. $p_h ull$ is labeled u and v for the two faces of $\mathcal{D}(T,S)$ adjacent to e_1 and e_2. Note that these faces are necessarily distinct by Proposition 6. Therefore the degree of p_{hull} increases by at most $2+2+1+0=5$ to at most 10.

This completes the proof of Theorem 2. □

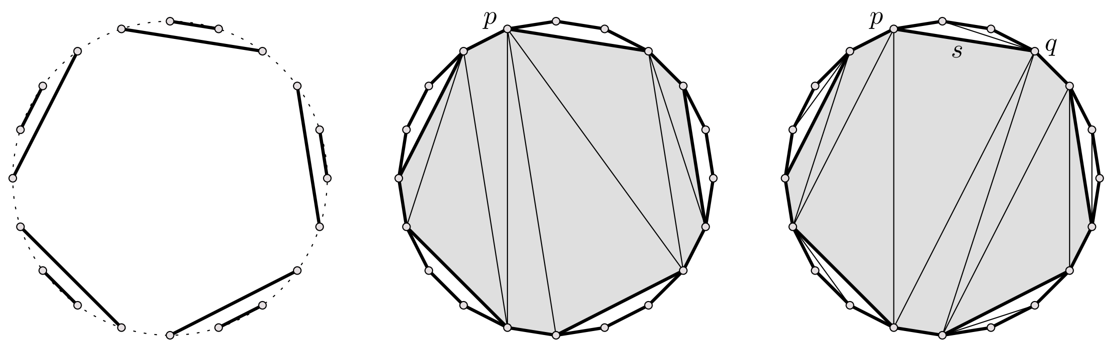


Figure 7: Lower bound construction (left), case 1 (middle), case 2 (right). D_{10} is shaded.

In the remainder of this section, we present a configuration of ten disjoint segments (planar straight line matching) such that for any pseudo-triangulation there is a vertex of degree at least 6. The segment endpoints form a convex 20-gon $P=(p_1,p_2,\ldots,p_{20})$. Five segments lie on the sides $p_{4k+2}p_{4k+3}$, and another five are on the diagonals $p_{4k+1}p_{4k+4}$ for $k=1,2,\ldots,5$, see Figure 7. The segments partition their convex hull into five convex quadrilaterals and a convex 10-gon D_{10}. Every vertex of the 10-gon is incident to a quadrilateral. Note that every pseudo-triangulation of a convex polygon is a triangulation. We distinguish two cases:

Case 1: A vertex p is incident to at least 3 diagonals of D_{10}. The degree of p is at least 5 in the triangulation of D_{10}. Together with the hull edge of the adjacent quadrilateral, the total degree of p is at least 6.

Case 2: Every vertex is incident to at most 2 diagonals of D_{10}. The diagonals of D_{10} form a path and so the dual graph of this triangulation is also a path. Among the 10 vertices of D_{10}, there are 2 consecutive pairs incident to less than 2 diagonals of D_{10}. Therefore, there are 3 consecutive vertices along D_{10} such that each of them is incident to 2 diagonals of D_{10}. So there is an input segment $s = (p, q)$ along D_{10} such that both its endpoints are incident to two diagonals of D_{10}. W.l.o.g., we may assume that q is also incident to a diagonal in the triangulation of the convex quadrilateral adjacent to s. The degree of q is 4 in D_{10} and 3 in the quadrilateral, so its total degree is six.

6 Conclusion

We derived a constant (vertex-)degree bound for pseudo-triangulations of disjoint line segments: There is always a pseudo-triangulation of (vertex-)degree at most ten. What is the smallest (vertex-)degree bound for pointed pseudo-triangulations of disjoint line segments in the plane?

A colorful extension of the result of Bose et al. [4] on binary encompassing trees was considered. A graph is *vertex-colored* if every vertex has a color and no edge is monochromatic. The set of input segments can be considered as a vertex-colored matching. Hurtado et al. [12] showed that every vertex-colored PSLG without singleton components admits a vertex-colored encompassing graph. Extending techniques of Hurtado et al. [12] and Hoffmann and Tóth[10], Souvaine and Tóth [20] have recently obtained an optimal degree bound for encompassing graphs of vertex-colored PSLGs. They can augment any vertex-colored PSLG with n vertices and no singleton components to a vertex-colored encompassing graph such that the degree of every vertex increases by at most two. On the other hand, Grantson, Meijer, and Rappaport [9] have recently shown that a vertex-colored *minimum* encompassing tree for a set of colored disjoint line segments may require a vertex of linear degree.

Our Theorem 1 does not generalize to PSLGs: By a result of Souvaine and Tóth [20], there is pointed PSLG such that it cannot be augmented to a *pointed* encompassing graph such that every vertex degree increases by at most two. A common generalization of the colorful encompassing trees and our Theorem 1 may still be possible: Can an encompassing tree *for disjoint segments* have all three properties combined: For a set of vertex-colored disjoint line segments in the plane, does there exist an encompassing tree such that (i) it has maximum degree three, (ii) it is pointed, and (iii) vertex-colored?

References

[1] P. K. Agarwal, J. Basch, L. J. Guibas, J. Hershberger, and L. Zhang, Deformable free space tilings for kinetic collision detection, in *Proc. 4th WAFR*, 2001, 83–96.

[2] O. Aichholzer, M. Hoffmann, B. Speckmann, and Cs. D. Tóth, Degree bounds for constrained pseudo-triangulations, in *Proc. 15th Canadian Conf. on Comput. Geom.*, (Halifax, NS, 2003) pp. 155–158.

[3] O. Aichholzer, C. Huemer, and H. Krasser. Triangulations without pointed spanning trees, in *Abstracts of 20th European Workshop Comput. Geom.*, 2004, pp. 221–224.

[4] P. Bose, M. E. Houle, and G. T. Toussaint, Every set of disjoint line segments admits a binary tree, *Discrete Comput Geom.* **26** (2001), 387–410.

[5] P. Bose and G. T. Toussaint, Growing a tree from its branches, *J. Algorithms* **19** (1995), 86–103.

[6] B. Chazelle, H. Edelsbrunner, M. Grigni, L. J. Guibas, J. Hershberger, M. Sharir, and J. Snoeyink, Ray shooting in polygons using geodesic triangulations, *Algorithmica* **12** (1994), 54–68.

[7] D. Eppstein, G. F. Italiano, R. Tamassia, R. E. Tarjan, J. Westbrook, and M. Yung, Maintenance of a minimum spanning forest in a dynamic plane graph, *J. Algorithms* **13** (1992), 33–54. Corrigendum in *J. Algorithms* **15** (1993), 173.

[8] M. Goodrich and R. Tamassia. Dynamic ray shooting and shortest paths in planar subdivision via balanced geodesic triangulations. *J. Algorithms 23* (1997), 51–73.

[9] M. Grantson, H. Meijer, and D. Rappaport. Bi-chromatic minimum spanning trees, in *Abstracts of 21st Euro. Workshop Comput. Geom.*, 2005, pp. 199–202.

[10] M. Hoffmann and Cs. D. Tóth, Pointed and colored binary encompassing trees, in *Proc. 21st Sympos. Comput. Geom.*, 2005, 81–90.

[11] M. Hoffmann and Cs. D. Tóth, Segment endpoint visibility graphs are Hamiltonian, *Comput. Geom. Theory Appl.* **26** (2003), pp. 47–68.

[12] F. Hurtado, M. Kano, D. Rappaport, and Cs. D. Tóth, Encompassing colored crossing-free geometric graphs, in Proc. *16th Canadian Conf. Comput. Geom.*, 2004, pp. 48–52.

[13] L. Kettner, D. Kirkpatrick, A. Mantler, J. Snoeyink, B. Speckmann, and F. Takeuchi, Tight degree bounds for pseudo-triangulations of points, *Comput. Geom. Theory Appl.* **25** (2003), 1–12.

[14] D. Kirkpatrick and B. Speckmann, Kinetic maintenance of context-sensitive hierarchical representations for disjoint simple polygons, in *Proc. 18th Sympos. Comput. Geom.*, 2002, pp. 179–188.

[15] D. T. Lee and A. K. Lin, Generalized Delaunay triangulations for planar graphs, *Discrete Comput. Geom.* **1** (1986), 201–217.

[16] M. Pocchiola and G. Vegter, Minimal tangent visibility graphs, *Comput. Geom. Theory Appl.* **6** (1996), 303–314.

[17] M. Pocchiola and G. Vegter, Topologically sweeping visibility complexes via pseudo-triangulations, *Discrete Comput. Geom.* **16** (1996), 419–453.

[18] D. Rappaport, Computing simple circuits from a set of line segments is NP-complete, *SIAM J. Comput.* **18** (1989), 1128–1139.

[19] D. Rappaport, H. Imai and G. T. Toussaint, Computing simple circuits from a set of line segments, *Discrete Comput. Geom.* **5** (1990), 289–304.

[20] D. Souvaine and Cs. D. Tóth, A vertex-face assignment for plane graphs, in *Proc. 17th Canadian Conf. Comput. Geom.*, (Windsor, ON, 2005), to appear.

[21] B. Speckmann and Cs. D. Tóth, Allocating vertex π-guards in simple polygons via pseudo-triangulations, *Discrete Comput. Geom.* **33** (2) (2005), 345–364.

[22] I. Streinu, A combinatorial approach to planar non-colliding robot arm motion planning, in *Proc. 41st Symp. Foundations Comp. Sci.*, IEEE Press, 2000, pp. 443–453.