Vertex-Colored Encompassing Graphs*

Michael Hoffmann† Csaba D. Tóth‡

October 16, 2010

Abstract

It is shown that every disconnected vertex-colored plane straight line graph with no isolated vertices can be augmented (by adding edges) into a connected plane straight line graph such that the new edges respect the coloring and the degree of every vertex increases by at most two. The upper bound for the increase of vertex degrees is best possible: there are input graphs that require the addition of two new edges incident to a vertex. The exclusion of isolated vertices is necessary: there are input graphs with isolated vertices that cannot be augmented to a connected vertex-colored plane straight line graph.

1 Introduction

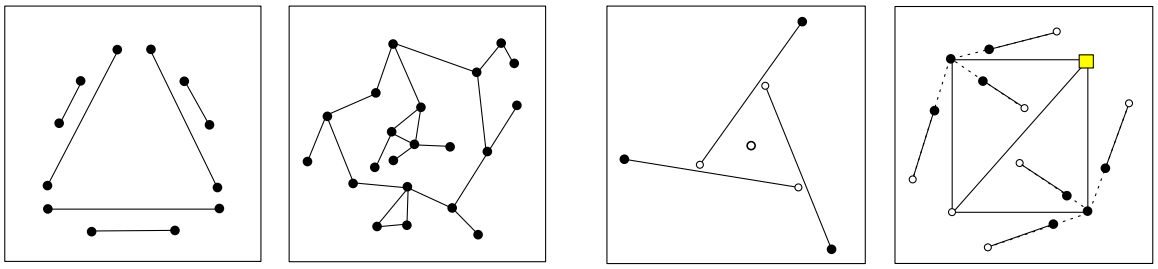
Spanning trees defined on disjoint objects in the plane are fundamental structures in computational geometry. Complex planar objects are often modeled by their boundary polygons which, in turn, can be represented as a plane straight line graph (PSLG). Consider a (possibly disconnected) PSLG $G = (V, E)$. If we add edges to G, such that we obtain a *connected* PSLG $G' = (V, E \cup E')$, then G' is an *encompassing graph* for G. Every PSLG admits an encompassing graph, the constrained Delaunay triangulation [15] is one of the well-known examples. In this paper, we address the problem of constructing an encompassing graph for a given PSLG such that each vertex is incident to a bounded number of new edges. For an empty graph, with n isolated vertices, an encompassing graph with the smallest maximum degree is a Hamiltonian path, in which the degrees of $n - 2$ vertices each increase from 0 to 2. The sparsest PSLG without isolated vertices is a perfect matching, which can be regarded as a set of disjoint line segments in the plane. A simple construction (Figure 1a) shows that some sets of disjoint line segments in the plane do not admit any encompassing path, and so every encompassing graph has a vertex of degree at least three. Bose *et al.* [4, 5] showed that every set of disjoint segments admits an encompassing tree of maximum degree three. Souvaine and Tóth [17] proved that every PSLG admits an encompassing graph such that the degree of every vertex increases by at most two.

In this paper, we impose further constraints on the encompassing graph to be constructed. Recall that a graph is called *(properly) vertex-colored* if every vertex has a color and adjacent vertices have different colors. For a vertex-colored graph, all edges of an encompassing graph must respect the coloring. Some vertex-colored graphs do not admit an encompassing graph:

*Preliminary results (on encompassing trees for vertex-colored plane straight line forests) have been published in the *Proceedings of the 21st ACM Symposium on Computational Geometry (Pisa, 2005)*, ACM Press, 2005, pp. 81–90.

†Institute of Theoretical Computer Science, ETH Zürich, CH-8092 Zürich, Switzerland, email: hoffmann@inf.ethz.ch

‡University of Calgary, Calgary, AB, Canada and Tufts University, Medford, MA, USA, email: cdtoth@cs.tufts.edu Research by Tóth was conducted at the Massachusetts Institute of Technology.



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

Figure 1: (a) Six segments that do not admit an encompassing path: in every encompassing tree, at least one segment endpoint is incident to at least two new edges. (b) A connected PSLG with 4 faces, including the outer face. (c) A vertex-colored PSLG with one isolated vertex and no encompassing graph. (d) If three vertices are collinear, then the degree of some vertices may have to increase by more than two.

For instance, no edge can be added to an empty graph with $n \geq 2$ isolated vertices of the same color. We will show that isolated vertices are essentially the only obstruction.

Theorem 1. *Every vertex-colored plane straight line graph with no isolated vertices and no three collinear vertices admits an encompassing graph such that the degree of each vertex increases by at most two.*

The assumptions about isolated vertices and collinear vertices cannot be dropped: Figure 1c-d depict vertex-colored PSLGs that do not admit encompassing graphs; the PSLG in Figure 1c has an isolated vertex, and the PSLG in Figure 1d has three collinear vertices. (In fact, our algorithm works correctly under a weaker non-collinearity condition: It is enough to assume that for every vertex v of degree one, the ray emitted by v along the supporting line of the incident edge does not hit any vertex of G.) Our proof for Theorem 1 is constructive, and we present a polynomial time algorithm to construct an encompassing graph. We also extend the results of Bose *et al.* [4, 5] to a vertex-weighted setting for disjoint line segments.

Corollary 2. *Let S be a finite set of disjoint line segments in the plane, where the two endpoints of every segment are labeled ① and ②, respectively. Then S admits an encompassing tree such that the degree of every vertex labeled ① increases by at most one, and the degree of every vertex labeled ② increases by at most two.*

The bound of at most three new edges per segment is best possible. Figure 1a depicts a set of disjoint line segments such that in every encompassing tree, there is an input segment incident to at least three new edges.

Related previous work. One of the simplest disconnected PSLGs is a finite set of disjoint line segments in the plane. A simple construction (Figure 1a) shows that not every finite set of disjoint segments in the plane admits an encompassing path. But there is always a path that encompasses $\Theta(\log n)$ segments and does not intersect any other input segment [8]. Also, there is always an encompassing graph that is Hamiltonian [9].

Bose, Houle, and Toussaint [4, 5] showed that every set of n disjoint line segments in the plane admits an encompassing tree of maximum degree *three*. The upper bound on the degree is best possible. This was generalized to arbitrary PSLGs by Souvaine and Tóth [17], they proved that every PSLG admits an encompassing graph such that the degree of every vertex increases by at most two, and it can be constructed in $O(n \log n)$ time.

Hoffmann, Speckmann, and Tóth [7] showed that every set of n disjoint segments in the plane admits a pointed binary encompassing tree, and it can be constructed in $O(n \log n)$ time. A vertex v of a PSLG is *pointed* if all incident edges lie in a halfplane whose boundary contains v; a PSLG is *pointed* if all its vertices are pointed.

Vertex-colored PSLGs and geometric graphs have also received considerable attention. In these problems, the input often consists of a set R of red points and a set B of blue points in the plane. A typical question asks whether the vertex set $R \cup B$ admits a vertex-colored PSLG with some special properties. Every vertex-colored graph is a subgraph of the complete bipartite geometric graph $K(R, B)$. A pioneer result in this area says that any n red and n blue points in the plane admit a vertex-colored plane straight line *matching* (for example, a minimum length bipartite matching is a plane graph).

Akiyama and Urrutia [1] constructed a configuration of n red and n blue points in the plane that does not admit any vertex-colored plane straight line Hamiltonian tour. Kaneko, Kano, and Yoshimoto [14] have found a colored point configuration for which every vertex-colored straight line Hamiltonian tour has at least $n - 1$ crossings. Kaneko and Kano [13] showed that if $|R| = \Theta(|B|^2)$ then all *blue* points can be covered by a vertex-colored plane straight line path.

Kaneko [11] proved that any set of n red and n blue points in the plane, no three of which are collinear, admits a vertex-colored plane straight line spanning tree with maximum degree three. Our Theorem 1 extends this result and shows that such a tree can encompass any given vertex-colored plane matching of the $2n$ input points. Hurtado *et al.* [10] proved that every vertex-colored plane straight line matching admits an encompassing tree. Theorem 1 extends their result and shows that such a tree exists with maximum degree three. For other results on geometric red-blue graphs, we refer the reader to the excellent survey by Kaneko and Kano [12].

Researchers have also studied the *minimum encompassing tree* (where the sum of edge lengths is minimal) for a set of n disjoint line segments in the plane. A minimum encompassing tree has maximum degree at most seven, which is the best possible bound, and it can be constructed greedily [5]. A *vertex-colored* minimum encompassing tree for a set of n bicolored disjoint line segments, however, may require a vertex of degree $\Omega(n)$ [3]. Moreover, if no two edges are allowed to cross, then it is NP-hard to find the minimum bicolored encompassing tree [2].

2 Preliminaries

Definitions. A *plane straight line graph* (for short, PSLG) G is a graph where the vertices are distinct points in the Euclidean plane, and the edges are straight line segments such that any two edges are disjoint with the possible exception of common endpoints. We denote by $V(G)$ and $E(G)$, respectively, the set of vertices and edges of G. In a slight abuse of notation, we also sometimes denote by G the planar point set covered by the vertices and edges of a PSLG G. The connected components of its complement $\mathbb{R}^2 \setminus G$ are called the *faces* of G. Denote the set of faces by $F(G)$. Note that faces are open sets and, therefore, the interior $\text{int}(f)$ of a face f is the same as f. Nevertheless we will occasionally write $\text{int}(f)$ to emphasize this fact.

A *corner* of a PSLG G is an ordered triple $c = (u, v, w)$ of three vertices of G such that uv and vw are edges in G, and they are consecutive in the counter-clockwise order of all edges incident to v. The *apex* of a corner $c = (u, v, w)$ is the vertex v, which we sometimes denote by $\text{a}(c) = v$. We say that a corner (u, v, w) is *incident* to its apex v. We also say that corner (u, v, w) is *incident* to a face $f \in F(G)$ if the vertex v is incident to f, and f lies to the left of both uv and vw. Hence every corner of G is incident to a unique face. We represent the boundary ∂f of a face f as a circular list $(p_1, p_2, \ldots, p_k)$ of vertices in the order in which the face lies to the left of each edge $p_i p_{i+1}$. In this representation, every triple (p_{i-1}, p_i, p_{i+1}) is a

corner of G incident to f. We say that a corner (u, v, w) is *convex* (*reflex*) if the angle that rotates vu to vw clockwise about v is less than (more than) 180°.

A plane straight line circuit is a *simple polygon*. Note that the boundary of a face of a PSLG G does not necessarily form a simple polygon, see for example the large bounded face in Figure 1b. However, the edges adjacent to a simply connected face of G form a *weakly simple polygon*, that is, a closed polygonal chain such that for every $\varepsilon > 0$, there is an ε-perturbation of the vertices that results in a simple polygon.

Overview of our algorithm. We compute an encompassing graph based on a recursive scheme developed by Hurtado *et al.* [10] (Algorithm 1 below). This scheme constructs an encompassing graph for a given vertex-colored PSLG, but it does not provide a bound on the number of new edges per vertex.

The input is a PSLG $G = G_0$ with no isolated vertex. At each step, the current graph G_i is augmented with a new edge that connects two components. Specifically, Hurtado *et al.* [10] define visibility such that the edges and vertices of G_i are opaque. An edge uv is *fully visible* from a corner c if the triangle $\triangle = \mathrm{a}(c)uv$ lies in the angular domain of c, and the interior of $\triangle$ is disjoint from G_i. They show that at each step, an edge in some component of G_i is fully visible from a corner c_i in another component. As G_i is vertex-colored, the two endpoints of the edge have different colors, and so one of them can be connected to $\mathrm{a}(c_i)$ such that the new edge respects the coloring.

Algorithm 1 (Hurtado *et al.* [10])**.**

Input: a vertex-colored PSLG *G.*

- *Let $i = 0$, let $G_0 = G$, and let A_0 be a component of G incident to the outer face of G.*
- *Repeat until G_i is connected;*
  1. *Find a pair (c_i, u_iv_i) such that c_i is a corner of A_i, u_iv_i is an edge in $G - A_i$, and u_iv_i is fully visible from c_i.*
  2. *If $\mathrm{a}(c_i)$ and u_i have different colors then let $G_{i+1} = G_i + \mathrm{a}(c_i)u_i$, otherwise let $G_{i+1} = G_i + \mathrm{a}(c_i)v_i$. Let A_{i+1} be the component of G_{i+1} containing A_0. Put $i := i+1$.*

Output G_i.

The above algorithm produces an encompassing graph, but the degree of a vertex may increase arbitrarily. In order to establish a bound on the maximum increase of vertex degrees, we cannot content ourselves with just *any* corner from which an edge in another component is fully visible. Instead, this corner-edge pair has to be chosen very carefully. We employ a charging scheme based on the following reformulation[1] of a result by Souvaine and Tóth [17].

Theorem 3 ([17])**.** *For every* PSLG *G and a designated* root *vertex $v_0 \in V(G)$, there is a set S of corners of G, called* stem *corners, that satisfies the conditions listed below. Let the* stem count *of a vertex v (for short, $\mathrm{stc}(v)$) be the number of stem corners incident to v plus the number of reflex corners incident to v.*

(i) every face of G is incident to a unique stem corner;

(ii) for every vertex v, we have $\mathrm{stc}(v) \leq 2$;

(iii) $\mathrm{stc}(v_0) \leq 1$.

[1] Souvaine and Tóth use the term *reduced $\star$-assignment*.

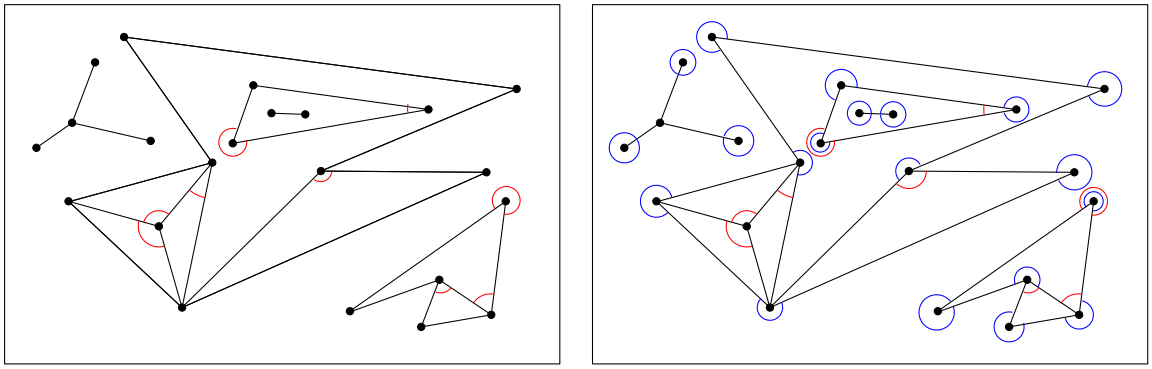


Figure 2: Left: a set S of stem corners for a PSLG. Right: the corners in S and all reflex corners (double arcs mark corners that are both).

Furthermore, a set of stem corners with these properties can be computed in $O(n \log n)$ time, where $n = |V(G)|$.

Note that if a stem corner is reflex, it is counted twice in $\mathrm{stc}(v)$. Hence, if a (non-root) vertex is incident to a reflex corner, then it is incident to at most one stem corner; if a (non-root) vertex is incident to convex corners only, then it is incident to at most two stem corners.

We say that a corner is *charged*, whenever a new edge is attached to its apex. In our charging scheme, each stem corner and each reflex corner corresponds to a possible new edge that may be attached to the apex of the corner. This will guarantee that the degree of a vertex v increases by at most $\mathrm{stc}(v) \leq 2$.

Whenever a new edge is attached to a vertex v of A_i, we *charge* this increase in the degree of v to one of the corners incident to v. Our goal is to ensure that each corner is charged at most once for being reflex and at most once for being stem. This will guarantee that the degree of every vertex increases by at most two. The details of the charging scheme are discussed in Section 4. Note that a new edge attached to a vertex v splits a corner incident to v into two corners. If this corner is reflex, then at most one of the two new corners is reflex. If this corner is a *stem* corner, then one of the two new corners will "inherit" the stem role as explained in Section 3.

3 Maintenance of stem corners

Throughout the algorithm, we maintain a component A_i of G_i along with a set S_i of stem corners for A_i. Intuitively, we "grow" the connected component A_i by attaching other components of G_i to it until $A_i = G_i$.

Preprocessing and initialization. For the sake of simplicity let us suppose that every component of $G - A_0$ lies in a bounded face of A_0. A PSLG does not have this property if the outer face is incident to several components. Therefore, we preprocess an input PSLG G^0 as follows.

Augment G^0 with a bounding box B, that is, four vertices and four edges forming a 4-cycle (Figure 3), with an arbitrary (proper) vertex coloring. Run the algorithm on the resulting graph G. The component A_0 of G adjacent to the outer face is B. We can choose an arbitrary convex corner of B to be the stem corner of its interior. Note that our algorithm inserts exactly one new edge between A_0 and G^0. Hence, we can remove B and this edge from the output and obtain an encompassing graph for G^0.

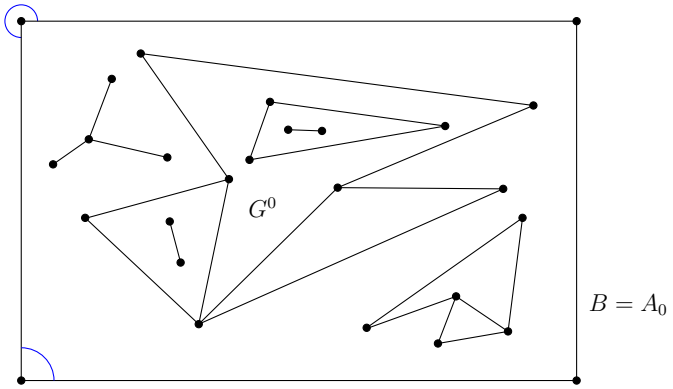


Figure 3: A PSLG G^0 with several components adjacent to the outer face, a bounding box B, and a set of stem corners for B.

Maintenance of faces and stem corners. After preprocessing, every component of $G_i - A_i$ lies in the interior of a bounded face of A_i. At Step i of the algorithm, suppose we are given the corner c_i and the edge u_iv_i that are to be connected. How exactly c_i and u_iv_i are chosen does not matter at this point and will be discussed later in Section 4. In any case, an edge u_iv_i of $G_i - A_i$ is fully visible from a stem corner or a reflex corner c_i of A_i.

Let f_i denote the face of A_i incident to c_i. Let $G_i \cap f_i$ denote the subgraph of G_i lying in the interior of f_i. Let D_i be the component of $G_i \cap f_i$ that contains edge u_iv_i. The new edge, say $\mathrm{a}(c_i)u_i$, connects components A_i and D_i of G_i. By attaching edge $\mathrm{a}(c_i)u_i$ and D_i to A_i, the face f_i is possibly decomposed into several faces: the bounded faces of D_i are carved out of f_i (see Figure 4). The remainder of f_i (lying in the outer face of D_i) is called the *successor* $\mathrm{succ}(f_i)$ of f_i. The area of the successors of a face is monotonically decreasing, and their boundary is (strictly) monotonically increasing, since $\mathrm{a}(c_i)u_i$ is a bridge of G_{i+1}.

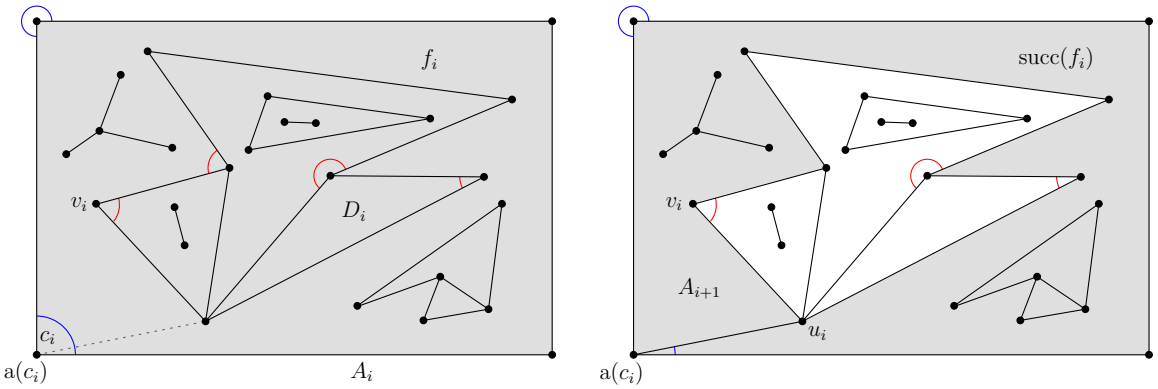


Figure 4: Left: The stem corners in S_i and in $S(D_i)$, with the face f_i shown gray. Right: The stem corners in S_{i+1}, with the face $\mathrm{succ}(f_i)$ shown gray.

We maintain a set S_i of stem corners for A_i as follows. Initially, start with an arbitrary set S_0 of stem corners for $A_0 = B$. Then construct the set S_{i+1} of stem corners for $A_{i+1} = A_i + \mathrm{a}(c_i)u_i + D_i$, essentially by combining S_i with an arbitrary set $S(D_i)$ of stem corners for D_i with u_i as a root.

Observe that the only face of G_{i+1} incident to some vertices of both A_i and D_i is $\mathrm{succ}(f_i)$. Any other face of G_{i+1} is incident to vertices of either A_i or D_i only. Let all these faces keep their stem corner. It remains to choose a stem corner for face $\mathrm{succ}(f_i)$.

If the stem corner of f_i in S_i is not c_i, then let it also be the stem corner of $\mathrm{succ}(f_i)$ in S_{i+1}. Otherwise, c_i is the stem corner of f_i in S_i. The new edge $\mathrm{a}(c_i)u_i$ splits c_i into two new corners, one on each side of $\mathrm{a}(c_i)u_i$. Let the one to the *right* of this edge be the stem corner of $\mathrm{succ}(f_i)$ in S_{i+1}. It is easy to verify that the set of stem corners for A_{i+1} obtained in this way satisfies the conditions listed in Theorem 3.

By choosing u_i to be the root for the set of stem corners for D_i, we account for incrementing the degree of u_i by one. Accounting for the increase of degree at $\mathrm{a}(c_i)$ turns out to be more involved and is the topic of the next section. The following properties of the set of stem corners should be kept in mind.

Proposition 4.

- *If the stem corner of f_i is not c_i, then* $\mathrm{succ}(f_i)$ *has the same stem corner as f_i.*
- *If c_i is the stem corner of f_i, then the new edge* $\mathrm{a}(c_i)u_i$ *splits c_i into two corners, and the one to the right of* $\mathrm{a}(c_i)u_i$ *is the stem corner of* $\mathrm{succ}(f_i)$. □

4 Charging edges to corners

In this section, we show how to choose a suitable corner-edge pair (c_i, u_iv_i) in Step i of Algorith 1. In particular, c_i should be a stem or reflex corner of A_i from which edge u_iv_i is fully visible. Furthermore, as the new edge—either $\mathrm{a}(c_i)u_i$ or $\mathrm{a}(c_i)v_i$—has to be charged to c_i, we must guarantee that every stem corner plays the role of c_i at most once, and similarly every reflex corner plays the role of c_i at most once. (Recall that the increase in degree at u_i or v_i is accounted for by making this vertex the root in the set of stem corners for the newly connected component.)

The new edge splits corner c_i into two new corners, so formally the corner c_i is destroyed. However, for a stem corner c_i exactly one of these new corners becomes a stem corner and for a reflex corner c_i at most one of the new corners is a reflex corner. In terms of the charging scheme, it is natural to identify the new stem and/or reflex corner with the stem and/or reflex corner c_i. Using this identification, we refrain from introducing a successor notation for corners.

Shortest paths in a face. Our approach builds on shortest paths in a simply connected face of a PSLG, described below. Shortest paths within a simple polygon are well-understood, cf. [6, 16]. In particular, any interior vertex of a shortest path is an apex of a reflex corner of the polygon. Also, two shortest paths starting from the same point do not cross, although their initial portions may overlap.

The faces in our graph G_i are weakly simple polygons that are not simple in general. However, if we specify not only a vertex of the face as a starting point for a shortest path but specify a corner of the face instead, then these shortest paths behave exactly the same as in the case of a simple polygon.

In Step i of the algorithm, we have a PSLG G_i and a component A_i incident to the outer face of G_i. If G_i is not connected, then every component of $G_i - A_i$ lies in some bounded face of A_i. Let f be a face of A_i, denote by s the stem corner of f in S_i, and denote by $G_i \cap f$ the subgraph of G_i lying in the interior of f. For a point $q \in \mathrm{int}(f)$, we denote by $\mathrm{p}_i(s, q)$ the *shortest path* between s and q that does not cross ∂f.

Accessible edges. Several edges of $G_i - A_i$ may be fully visible from a corner c_i of A_i. The choice of this corner is not arbitrary, since we want to ensure that if a new edge is charged to c_i, then c_i will not be charged again later. In our algorithm, we always choose the corner c_i

and the edge u_iv_i such that $\mathrm{a}(c_i)u_i$ or $\mathrm{a}(c_i)v_i$ lies on the shortest path from the stem corner to u_i or v_i, respectively. Furthermore, we will always choose c_i such that the shortest path from the stem corner to c_i does not intersect $G_i \cap f$. Intuitively, this means that we give priority to corners that are closer to the stem (measured by the shortest distance within the face).

Let e be an edge of $G_i - A_i$. Let f denote the face of A_i containing e, and let s be the stem corner of f in S_i. Edge e is *accessible* from a corner c of f, if

(i) the edge e is fully visible from c,

(ii) the shortest path $\mathrm{p}_i(s, c)$ is disjoint from $G_i - A_i$,

(iii) for an endpoint q of e, the shortest path $\mathrm{p}_i(s, q)$ is incident to c.

The following observation is immediate.

Proposition 5. *Consider an edge e of $G_i - A_i$, and let s denote the stem corner of the face of A_i that contains e. Then e is accessible from s if and only if e is fully visible from s.* □

We show next that some edge of $G_i - A_i$ is always accessible unless G_i is connected.

Lemma 6. *Unless $G_i = A_i$, some edge e of $G_i - A_i$ is accessible from some corner of the face of A_i that contains e.*

Proof. Since $G_i \neq A_i$, there is some bounded face f of A_i that contains some component of G_i, that is, $G_i \cap \mathrm{int}(f) \neq \emptyset$.

From the apex of every reflex corner of $G_i \cap f$ shoot a ray collinear to one of the incident edges of $G_i \cap f$. In this way the reflex angle is decomposed into two convex angles. Draw a directed line segment (called *extension*) along each ray successively from the apex until it hits ∂f, an edge of $G_i \cap f$, or a previous extension (Figure 5). The graph $G_i \cap f$ and the extensions jointly decompose f into *regions*. Since no three vertices are collinear, the stem corner s of f is incident to a unique region, which we denote by R.

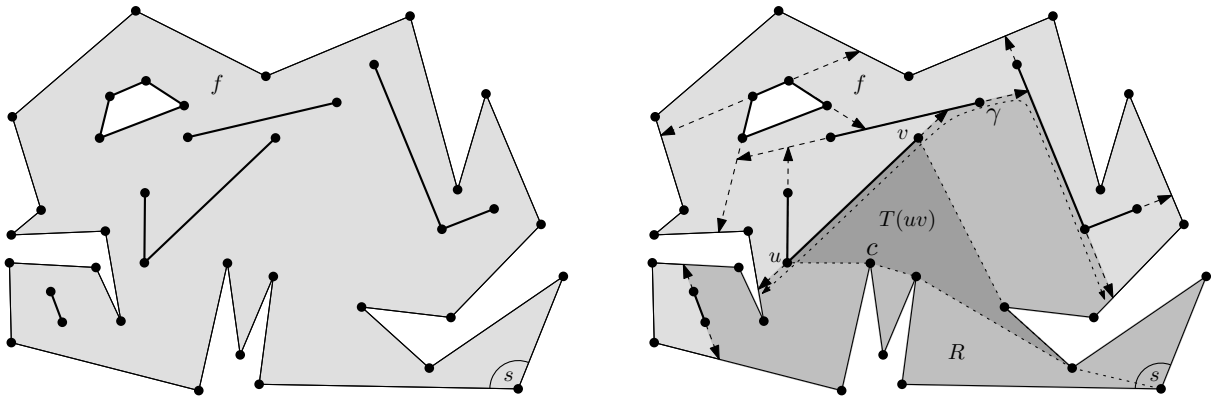


Figure 5: Left: a face $f \in F(G_i)$ and the components $G_i \cap F_i$ lying in f_i. Right: the decomposition of a face f, the region R incident to the stem corner s of f, a curve γ along ∂R (slightly offset to increase readability), a complete edge uv along $\gamma \subset \partial R$, and the corresponding pseudo-triangle $T(uv)$.

We show that ∂R contains an entire edge of $G_i \cap f$. The boundary ∂R is composed of parts of ∂f, parts of $G_i \cap f$, and parts of some extensions. Consider a connected component γ of $\partial R - \partial f$: γ is a polygonal path whose endpoints lie on ∂f. If we traverse γ in counterclockwise orientation with respect to R, the first and the last edge of γ are extensions directed towards the endpoints

of γ. Let $u \in \gamma$ be the first point along γ incident to an edge directed counterclockwise. Point u is a vertex of G_i which emits an extension in counterclockwise direction. It is incident to some edge $e = uv$ of $G_i \cap f$ that lies on the boundary of R.

By construction, all reflex corners of R are located at reflex corners of f. It follows that a shortest path between any two points in R with respect to R is also a shortest path with respect to f. In particular, the shortest path (with respect to f) between s and any point along γ lies in R.

Let $T(uv)$ denote the simple[2] polygon enclosed by uv and the two shortest paths from s to u and v, respectively (Figure 5). It is known [16] that $T(uv)$ is a pseudo-triangle, that is, a simple polygon with exactly three convex corners. Two convex corners of $T(uv)$ are at u and v, a third convex corner is at a stem or reflex corner of f, and all other corners of $T(uv)$ are at reflex corners of f.

Consider an arbitrary triangulation of $T(uv)$. Let c be the corner of f where the third vertex of the triangle incident to e (other than u and v) is located. Then c is a stem or reflex corner of f, the edge e is fully visible from c, and c is on the shortest path from s to at least one of u or v. Since both u and v lie in R, the path $\mathrm{p}_i(s, c)$ is disjoint from $G_i - A_i$. Altogether, it follows that the edge uv is accessible from c. □

Extensible and safe corners. The following definitions provide some kind of depth order for the corners of f with respect to the stem corner. A corner c of f is *extensible* if and only if there exists some edge e of $G_i - A_i$ that is accessible from c. If a corner is not extensible we refer to it as *non-extensible*. A corner c of f is *safe* if and only if all corners other than c on the shortest path $\mathrm{p}_i(s, c)$ from the stem corner s of f to c are non-extensible. A corner that is both safe and (non-)extensible is referred to as *safely (non-)extensible*.

Choosing corner and edge. In Step i of the algorithm, we pick an (arbitrary) corner c_i of some face f_i that is safely extensible. By Lemma 6 there is an extensible corner and, therefore, there is also a safely extensible corner in G_i. For instance, if the stem corner s of f_i is extensible, then $c_i = s$. It remains to pick an edge $u_i v_i$ accessible from c_i. We distinguish two cases.

> **Case 1:** $c_i = s = (a_0, a_1, a_2)$ is the stem corner of f_i. Let $u_i v_i$ be the edge accessible from s for which the angle $\angle(a_2, a_1, u_i)$ is minimal.
>
> **Case 2:** $c_i \neq s$ is a reflex corner of f_i. Consider the set U of all endpoints q of edges accessible from c_i for which the shortest path $\mathrm{p}_i(s, q)$ passes through c_i. Let u_i denote a point from U for which $\mathrm{p}_i(s, u_i)$ makes an angle closest to 180° at c_i. Choose v_i correspondingly such that $u_i v_i$ is an edge that is accessible from c_i.

This finishes the description of our algorithm. The following proposition justifies the term "safe" by establishing some kind of life cycle of a corner that will turn out to be essential for our charging scheme.

Proposition 7. *Suppose that at begin of Step i of the algorithm, s is the stem corner of a face f of A_i, a corner c of f is safely non-extensible, and $\mathrm{p}_i(s, c)$ is disjoint from $G_i - A_i$. Then c remains safely non-extensible in the remainder of the algorithm.*

Proof. Consider a corner c that is safely non-extensible at begin of Step i of the algorithm, and $\mathrm{p}_i(s, c)$ is disjoint from $G_i - A_i$. We prove by induction on $|\mathrm{p}_i(s, c)|$, the number of vertices of path $\mathrm{p}_i(s, c)$, that c remains safely non-extensible and $\mathrm{p}_i(s, c) = \mathrm{p}_j(s, c)$, for all $j \geq i$.

[2]Strictly speaking, the shortest paths $\mathrm{p}_i(s, u)$ and $\mathrm{p}_i(s, v)$ may share an initial portion (Figure 5). We do not consider such a common prefix to be part of $T(uv)$ in order to obtain a simple polygon.

The base case is $|\mathrm{p}_i(s,c)| = 1$, that is, $s = c$. By Proposition 5, s is non-extensible if and only if no edge of $G_i \setminus A_i$ is fully visible from s. If no edge is fully visible from a corner, this property continues to hold throughout the algorithm, since A_ℓ increases with ℓ and visibility only decreases when the graph is augmented with new edges. Therefore s remains safely non-extensible, as claimed, and obviously the trivial path $\mathrm{p}_i(s,s)$ cannot change.

Assume that $\mathrm{p}_i(s,c) = (s = d_1, \ldots, d_k = c)$, for some $k > 1$, and consider some corner d_ℓ, $1 \leq \ell < k$. Since c is safe, d_ℓ is safely non-extensible at begin of Step i and, thus, by the inductive hypothesis d_ℓ remains safely non-extensible and $\mathrm{p}_j(s,d_\ell) = \mathrm{p}_i(s,d_\ell)$, for all $j \geq i$.

According to the definition of accessibility, there are three possible reasons why c is non-extensible at begin of Step i:

(1) either no edge of $G_i - A_i$ is fully visible from c;

(2) or $\mathrm{p}_i(s,c)$ crosses some edge of $G_i - A_i$;

(3) or $\mathrm{p}_i(s,c)$ is disjoint from $G_i - A_i$ but for every edge uv of $G_i - A_i$ fully visible from c, neither $\mathrm{p}_i(s,u)$ nor $\mathrm{p}_i(s,v)$ passes through c.

As argued for the base case, once (1) holds, it continues to hold throughout the algorithm. Otherwise—as (2) does not hold at begin of Step i by assumption—it is enough to show that (3) continues to hold throughout the algorithm.

Let $j > i$ be the smallest index such that at begin of Step j of the algorithm either (3) does not hold or $\mathrm{p}_j(s,c) \neq \mathrm{p}_i(s,c)$. Clearly, c is non-extensible between Step i and Step $(j-1)$. In particular, corner c still exists at begin of Step j.

Suppose that $\mathrm{p}_j(s,c) \neq \mathrm{p}_{j-1}(s,c)$. Since $\mathrm{p}_j(s,d_\ell) = \mathrm{p}_i(s,d_\ell)$, for all $1 \leq \ell < k$, the only way to change $\mathrm{p}_{j-1}(s,c)$ is to insert an edge that crosses $\mathrm{a}(d_{k-1})\mathrm{a}(c)$. In Step j of the algorithm, u_jv_j is accessible from c_j, and so $\mathrm{p}_j(s,u_j)$ or $\mathrm{p}_j(s,v_j)$ passes through c_j. Suppose w.l.o.g. that $\mathrm{p}_j(s,u_j)$ passes through c_j. Note that $\mathrm{a}(c_j)u_j$ is the last edge of the path $\mathrm{p}_j(s,u_j)$. If edge $\mathrm{a}(c_j)u_j$ is inserted in Step j, then it cannot cross $\mathrm{a}(d_{k-1})\mathrm{a}(c)$ as two shortest paths from the same source do not cross each other. Now suppose that edge $\mathrm{a}(c_j)v_j$ is inserted in Step j. Since u_jv_j is accessible from c_j, it is fully visible from c_j. In particular, the interior of the triangle $\mathrm{a}(c_j)u_jv_j$ is disjoint from G_j. Therefore, any path that crosses $\mathrm{a}(c_j)v_j$ must also cross u_jv_j or $\mathrm{a}(c_j)u_j$. We already observed that $\mathrm{p}_j(s,c)$ does not cross $\mathrm{a}(c_j)u_j$, so $\mathrm{p}_j(s,c)$ crosses u_jv_j. We have assumed that $\mathrm{p}_{j-1}(s,c) = \mathrm{p}_i(s,c)$. Since $G_i - A_i$ is monotonically decreasing, this implies that $\mathrm{p}_i(s,c)$ intersects an edge of $G_i - A_i$ at Step i already, contrary to our assumption that $\mathrm{p}_i(s,c)$ is disjoint from $G_i - A_i$. Therefore $\mathrm{p}_j(s,c) = \mathrm{p}_{j-1}(s,c) = \mathrm{p}_i(s,c)$.

It remains to consider the case that (3) does not hold at begin of Step j, that is, there is an edge uv of $G_j - A_j$ that is fully visible from c and, say, c lies on the path $\mathrm{p}_j(s,u)$. By the uniqueness of shortest paths it follows $\mathrm{p}_j(s,u) = (s = d_1, \ldots, d_k = c, u)$. As $\mathrm{p}_j(s,c) = \mathrm{p}_i(s,c)$ and visibility is monotonically decreasing during the algorithm, edge uv is fully visible from c in Step i already. Furthermore, we claim that c also appears along $\mathrm{p}_i(s,u)$ at begin of Step i already. Indeed, given that c appears as a reflex corner along $(s = d_1, \ldots, d_k = c, u)$ at Step j and that the prefix $(s = d_1, \ldots, d_k = c)$ of this path forms $\mathrm{p}_i(s,c)$ at both Step i and Step j, we conclude that the pseudo-triangle $T(c,u)$ (at both steps) degenerates into the line segment $\mathrm{a}(c)u$. Therefore our claim holds and so c is extensible at begin of Step i, contrary to our assumption. We may thus conclude that (3) continues to hold throughout the algorithm. □

Note that it is in general not true that every non-extensible corner remains non-extensible, because any change along the shortest path to the stem corner may affect extensibility.

Charging scheme. We will now argue that throughout the algorithm every corner is charged at most once for being reflex and at most once for being a stem corner of some face. As far as being a stem corner is concerned, this is not hard to see.

Proposition 8. *Every stem corner is charged at most once for being stem.*

Proof. Suppose that corner c_i, which is charged in Step i of the algorithm, is a stem corner. Denote by f_i the face of A_i incident to c_i. The insertion of $\mathrm{a}(c_i)u_i$ or $\mathrm{a}(c_i)v_i$ splits c_i into two corners. By Proposition 4, the corner c_i' to the right of $\mathrm{a}(c_i)u_i$ or $\mathrm{a}(c_i)v_i$, respectively, is the stem corner of $\mathrm{succ}(f_i)$. By choice of u_iv_i (as the first edge accessible from c that is swept), no edge inside $\mathrm{succ}(f_i)$ is fully visible from c_i'. Since visibility can only decrease under edge insertions, no edge of another component becomes fully visible and, thus, accessible from c_i' in the successors of f_i. In particular, c_i' remains the stem corner of these faces in the remainder of the algorithm. □

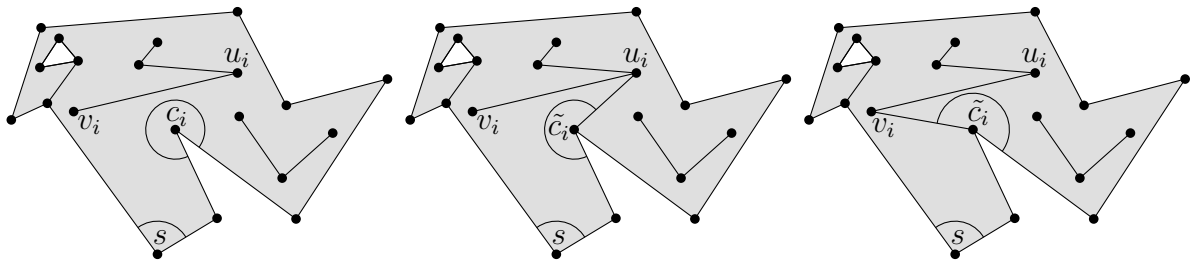
It remains to consider the case that $c_i \neq s$ is a reflex corner of f_i. The following proposition asserts the correctness of our charging scheme and thereby completes the proof of Theorem 1.

Proposition 9. *Every reflex corner is charged at most once for being reflex.*

Proof. Suppose that corner c_i, charged in Step i of the algorithm, is a reflex corner and not a stem corner. Denote by f_i the face of A_i incident to c_i, and let s be the stem corner of f_i. By Proposition 8, s remains the stem corner of f_i and all its successors in the remainder of the algorithm.

If c_i is split into two convex corners in Step i, then there is nothing to show. Hence suppose that c_i is split into two corners, one of which, denoted $\tilde{c}_i$, is reflex. We distinguish two cases.

First suppose that $\mathrm{p}_i(s, c_i) = \mathrm{p}_{i+1}(s, \tilde{c}_i)$, as shown in Figure 6b. Then by the choice of u_iv_i (as the "most straight" extension option for c_i), all other edges in G_i possibly accessible from c_i lie outside the angular wedge spanned by $\tilde{c}_i$. Therefore, $\tilde{c}_i$ is non-extensible in G_{i+1}. As $\mathrm{p}_i(s, c_i) = \mathrm{p}_{i+1}(s, \tilde{c}_i)$, the corner $\tilde{c}_i$ is safe and $\mathrm{p}_{i+1}(s, \tilde{c}_i)$ does not cross any edge of $G_{i+1} - A_{i+1}$. Thus by Proposition 7 it follows that $\tilde{c}_i$ remains safely non-extensible for the remainder of the algorithm, and so no further edge will be attached to it.



(a) Before Step i. (b) $\mathrm{p}_i(s, c_i) = \mathrm{p}_{i+1}(s, \tilde{c}_i)$. (c) $\mathrm{p}_i(s, c_i) \neq \mathrm{p}_{i+1}(s, \tilde{c}_i)$.

Figure 6: Possible positions of $\tilde{c}_i$.

Otherwise, $\mathrm{p}_i(s, c_i) \neq \mathrm{p}_{i+1}(s, \tilde{c}_i)$, as shown in Figure 6c. In this case, the new edge must be $\mathrm{a}(c_i)v_i$ and c_i does not lie on $\mathrm{p}_i(s, v_i)$. Since u_iv_i is accessible from c_i at the begin of Step i, c_i lies on $\mathrm{p}_i(s, u_i)$ and the interior of the triangle $\mathrm{a}(c_i)u_iv_i$ is disjoint from G_i. Moreover, $u_iv_i\mathrm{a}(c_i)$—the corner opposite to c_i in this triangle—is convex and is not a stem corner by Proposition 4 (and, independently, Proposition 8). Thus $\tilde{c}_i$—although being reflex—behaves essentially like a convex corner of f. In particular, it will never appear on any shortest path from s to any corner

of $G_j - A_j$, $j > i$. Therefore, no further edge will be attached to $\tilde{c}_i$ during the remainder of the algorithm. □

5 Weighted disjoint line segments

We prove Corollary 2 by slightly modifying our algorithm, as described below.

Corollary 2. *Let M be a plane straight line matching, where the two endpoints of every edge are labeled ① and ②, respectively. Then M admits an encompassing tree such that the degree of every vertex labeled ① increases by at most one, and the degree of every vertex labeled ② increases by at most two.*

Proof. Consider a plane straight line matching M, where the two endpoints of every edge are labeled ① and ②, respectively. Recall that during initialization our algorithm adds a bounding rectangle B, and sets $G_0 = M \cup B$, and $A_0 = B$. Every component of $G_0 - A_0$ is a line segment lying in the interior of B. In Step i, our algorithm finds a corner c_i of A_i from which an edge u_iv_i in a component of $G_i - A_i$ is fully visible. Note that u_iv_i is a line segment with one endpoint labeled ① and one endpoint labeled ②. Suppose without loss of generality that u_i is labeled ① and v_i is labeled ②. We modify the definition of A_{i+1} such that we set $G_{i+1} = G_i + \mathrm{a}(c_i)v_i$. Then A_{i+1} has no new faces, and two new reflex corners at $\mathrm{a}(u_i)$ and at $\mathrm{a}(v_i)$, respectively. It follows from the charging scheme that the degree of every vertex ① increases by at most one, and the degree of every vertex ② increases by at most two. □

Acknowledgment

We thank David Rappaport for useful insights into visibility sweep algorithms, and for many helpful comments on an earlier version of this paper.

References

[1] J. Akiyama and J. Urrutia, Simple alternating path problem, *Discrete Math.* **84** (1990), 101–103.

[2] M. G. Borgelt, M. van Kreveld, M. Löffler, J. Luo, D. Merrick, R. I. Silveira, and M. Vahedi. Planar bichromatic minimum spanning trees, *J. Discrete Alg.* **7** (2009), 469–478.

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

[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] L. J. Guibas, J. Hershberger, D. Leven, M. Sharir, R. E. Tarjan, Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons, *Algorithmica* **2** (1987), 209–233.

[7] M. Hoffmann, B. Speckmann, and Cs. D. Tóth, Pointed binary encompassing trees: simple and optimal, *Comput. Geom. Theory Appl.* **43** (2010), 35–41.

[8] M. Hoffmann and Cs. D. Tóth, Alternating paths through disjoint line segments, *Inf. Proc. Letts.* **87** (2003), 287–294.

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

[10] F. Hurtado, M. Kano, D. Rappaport, and Cs. D. Tóth, Encompassing colored crossing-free geometric graphs, *Comput. Geom. Theory Appl.* **39** (2008), 14–23.

[11] A. Kaneko, On the maximum degree of bipartite embeddings of trees in the plane, in *Discrete and Computational Geometry (Akiyama* et al.*, eds.)*, Japan Conference on Discrete and Computational Geometry 1998, vol. 1763 of LNCS, Springer, 2000, pp. 166–171.

[12] A. Kaneko and M. Kano, Discrete geometry on red and blue points in the plane—a survey, in *Discrete and Computational Geometry, The Goodman-Pollack Festschrift*, vol. 25 of Algorithms and Combinatorics, Springer, 2003, pp. 551–570.

[13] A. Kaneko and M. Kano, On paths in a complete bipartite geometric graph, in *Discrete and Computational Geometry (Akiyama* et al.*, eds.)*, Japan Conference on Discrete and Computational Geometry 2000, vol. 2098 of LNCS, Springer, 2001, pp 187–191.

[14] A. Kaneko, M. Kano, and K. Yoshimoto, Alternating Hamiltonian cycles with minimum number of crossings in the plane, *Internat. J. Comput. Geom. Appl.* **10** (2000), 73–78.

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

[16] D. T. Lee and F. P. Preparata, Euclidean shortest path in the presence of rectilinear barriers, *Networks* **14** (1984), 393–410.

[17] D. L. Souvaine and Cs. D. Tóth, A vertex-face assignment for plane graphs, *Comput. Geom. Theory Appl.* **42** (5) (2009), 388–394.