Reconstructing Generalized Staircase Polygons with Uniform Step Length

Nodari Sitchinava[1] and Darren Strash[2(✉)]

[1] Department of Information and Computer Sciences, University of Hawaii, Manoa, USA
nodari@hawaii.edu
[2] Department of Computer Science, Colgate University, Hamilton, NY, USA
dstrash@cs.colgate.edu

Abstract. *Visibility graph reconstruction*, which asks us to construct a polygon that has a given visibility graph, is a fundamental problem with unknown complexity (although visibility graph recognition is known to be in PSPACE). We show that two classes of uniform step length polygons can be reconstructed efficiently by finding and removing rectangles formed between consecutive convex boundary vertices called tabs. In particular, we give an $O(n^2m)$-time reconstruction algorithm for orthogonally convex polygons, where n and m are the number of vertices and edges in the visibility graph, respectively. We further show that reconstructing a monotone chain of staircases (a histogram) is fixed-parameter tractable, when parameterized on the number of tabs, and polynomially solvable in time $O(n^2m)$ under reasonable alignment restrictions.

Keywords: Visibility graphs · Polygon reconstruction
Visibility graph recognition · Orthogonal polygons
Fixed-parameter tractability

1 Introduction

Visibility graphs, used to capture visibility in or between polygons, are simple but powerful tools in computational geometry. They are integral to solving many fundamental problems, such as routing in polygons, and art gallery and watchman problems, to name a few. Efficient, and even worst-case optimal, algorithms exist for computing a visibility graph from an input polygon [16]; however, comparatively little is known about the reverse direction: the so-called visibility graph *recognition* and *reconstruction* problems.

In this paper, we study *vertex-vertex visibility graphs*, which are formed by visibility between pairs vertices of a polygon. Given a graph $G = (V, E)$, the visibility graph recognition problem asks if G is the visibility graph of *some*

polygon. Similarly, the visibility graph reconstruction problem asks us to construct a polygon with G as a visibility graph. Surprisingly, recognition of simple polygons is only known to be in PSPACE [13], and it is still unknown if simple polygons can be reconstructed in polynomial time. Therefore, current solutions are typically for restricted classes of polygons.

1.1 Special Classes

A well-known result due to ElGindy [11] is that every maximal outerplanar is a visibility graph and a polygon can be reconstructed from every such graph in polynomial time. Other special classes rely on a unique configuration of reflex and convex chains, which restrict visibility. For instance, spiral polygons [14], and tower polygons [7] (also called funnel polygons), can be reconstructed in linear time, and each consists of one and two reflex chains, respectively. 2-spirals can also be reconstructed in polynomial time [3], as can a more general class of visibility graphs related to 3-matroids [4].

For monotone polygons, Colley [8,9] showed that if each face of a maximal outerplanar graph is replaced by a clique on the same number of vertices, then the resulting graph is a visibility graph of some uni-monotone polygon (monotone with respect to a single edge), and such a polygon can be reconstructed if the Hamiltonian cycle of the boundary edges is known. However, not every uni-monotone polygon (even those with uniformly spaced vertices) has such a visibility graph [12]. Finally, Evans and Saeedi [12] characterized terrain visibility graphs, which consist of a single monotone polygonal line.

For orthogonal polygons, *orthogonal convex fans* (also known as *staircase polygons*), which consist of a single staircase and an extra vertex, can be recognized in polynomial time [2]; however—strikingly—the *only* class of orthogonal polygons known to be reconstructible in polynomial time is the staircase polygon with uniform step lengths, due to Abello and Eğecioğlu [1]. Other algorithms for orthogonal polygons use different visibility representations such as vertex-edge or edge-edge visibility [18, Sect. 7.3], or "stabs" [17]. See Asano et al. [5] or Ghosh [15] for a thorough review of results on visibility graphs.

1.2 Our Results

In this work, we investigate reconstructing polygons consisting of multiple uniform step length staircases. We first show that orthogonally convex polygons can be reconstructed in time $O(n^2 m)$. We further show that reconstructing orthogonal uni-monotone polygons is fixed-parameter tractable, when parameterized on the number of the horizontal convex-convex boundary edges in the polygon. We also provide an $O(n^2 m)$ time algorithm under reasonable alignment assumptions. As a consequence of our reconstruction technique, we can also recognize the visibility graphs of these classes of polygons with the same running times.

2   Preliminaries

Let P be a polygon on n vertices. We say that a point p *sees* a point q (or p and q are *visible*) in polygon P if the line segment pq does not intersect the exterior of P. Under this definition, visibility is allowed along edges and through vertices.

For our visibility graph discussion, we adopt standard notation for graphs and polygons. In particular, for a graph $G = (V, E)$, we denote the *neighborhood* of a vertex $v \in V$ by $N(v) = \{u \mid (v, u) \in E\}$, and denote the number of vertices and edges by $n = |V|$ and $m = |E|$, respectively. For a visibility graph $G_P = (V_P, E_P)$ of a polygon P, we call an edge in G_P that is an edge of P a *boundary edge*. Other edges (diagonals in P) are *non-boundary edges*.

Finally, critical to our proofs is the fact that a maximal clique in G_P corresponds to a maximal (in the number of vertices) convex region $R \subseteq P$ whose vertices are defined by vertices of P. A vertex v is called *simplicial* if $N(v)$ forms a clique, or equivalently v is in exactly one maximal clique. For our work here, we further adapt this definition for an edge. We say that an edge (u, v) is *1-simplicial* if $N(u) \cap N(v)$ is a clique, or equivalently (u, v) is in exactly one maximal clique[1]. The intuition behind why we consider 1-simplicial edges is that, in orthogonal polygons with edges of uniform length, boundary edges between convex vertices are 1-simplicial, with the vertices of the clique forming a rectangle. (See Fig. 1.)
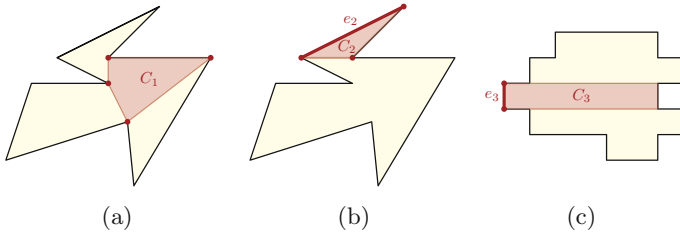

Fig. 1. Maximal convex regions on vertices of polygons are maximal cliques in visibility graphs. (b)–(c) 1-simplicial edges are in exactly one maximal clique.

Our running times depend on the following observation for 1-simplicial edges.

Observation 1. *We can test if (u, v) is 1-simplicial and in a maximal k-clique in time $O(kn)$.*

3   Uniform-Length Orthogonally Convex Polygons

We first turn our attention to a restricted class of orthogonal polygons that have only uniform-length (or equivalently, unit-length) edges. Let P be an orthogonal

---

[1] This is not to be confused with *simplicial edges*, which are defined elsewhere to be edges (u, v) such that for every $w \in N(u)$ and $x \in N(v)$, w and x are adjacent.

polygon with uniform-length edges such that no three consecutive vertices on P's boundary are collinear, and further let P be *orthogonally convex*[2]. We call P a *uniform-length orthogonally convex polygon* (UP). Note that every vertex v_i on P's boundary is either convex or reflex. We call boundary edges between two convex vertices in a uniform-length orthogonal polygon P *tabs* and a tab's endvertices *tab vertices*. We reconstruct the polygon by computing the clockwise ordering of vertices of the UP.

Note that the boundary of a UP consists of four tabs connected via staircases. For ease of exposition, we imagine the UP embedded in $\mathbb{R}^2$ with polygon edges axis-aligned. We call the tab with the largest y-coordinate the north tab, and we similarly name the others the south, east, and west tabs. We similarly refer to the four boundary staircases as northwest, northeast, southeast, and southwest.

We only consider polygons with more than 12 vertices, which eliminates many special cases. Smaller polygons can be solved in constant time via brute force.

We first introduce several structural lemmas which help us identify convex vertices in a UP, which is key to our reconstruction.

Lemma 1. *For every convex vertex u in a UP there is a convex vertex v, such that $(u, v) \in E_P$ and (u, v) is 1-simplicial.*

Proof. If u is a tab vertex, then the other tab vertex v is also convex and (u, v) is 1-simplicial. Otherwise, without loss of generality, suppose that u is on the northwest staircase. Then there is a convex vertex v on the southeast staircase that is visible from u. Edge (u, v) is in exactly one maximal clique, consisting of u, v, the reflex vertices within the rectangle R defined by u and v as the opposite corners, and any other corners of R that are convex vertices of the polygon. ⊠

Lemma 2. *In a UP, if u or v is a reflex vertex, then edge (u, v) is not 1-simplicial.*

Proof (Sketch[3]). If both u and v are reflex, then (u, v) is in one maximal clique consisting of only reflex vertices and another one that includes some convex vertex w. If one of u or v is convex, there exist two convex vertices w and w', forming two distinct maximal cliques with (u, v). ⊠

Lemma 2 states that only edges between convex vertices can be 1-simplicial. Hence it allows us to identify all convex vertices, by checking for each edge (u, v) if $N(u) \cap N(v)$ is a clique in $O(n^2)$ time, leading to the following lemma.

Lemma 3. *We can identify all convex and reflex vertices in a visibility graph of a UP in $O(n^2 m)$ time.*

We say a UP is *regular* if each of its staircase boundaries have the same number of vertices. Otherwise, we call it *irregular*, consisting of two *long* and two *short* staircases. We restrict our attention to irregular uniform-length orthogonally convex polygons (IUPs); however, similar methods work for their regular counterparts.

---

[2] That is, any two points in P can be connected by a staircase contained in P.

[3] Full proofs may be found in the full version of this paper [19].

3.1   Irregular Uniform-Length Orthogonally Convex Polygons

Let G_P be the visibility graph of IUP P. Our reconstruction algorithm first computes the four tabs, then assigns the convex and reflex vertices to each staircase. The following structural lemma helps us find the tabs. We assume that we have already computed the convex and reflex vertices in $O(n^2m)$ time.
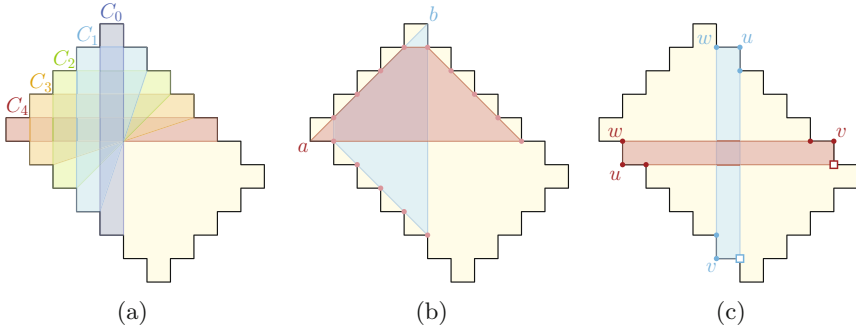

Fig. 2. Elements of our reconstruction. (a) Elementary cliques $C_0, \ldots, C_4$ interlock along a short staircase. (b) Tab vertices a and b see unique reflex vertices on long staircases. (c) Reflex vertices (square) are discovered by forming rectangles with known vertices u, v and w.

Lemma 4. *In every IUP there are exactly four 7-vertex maximal cliques, each containing exactly three convex vertices. Each such clique contains exactly one tab, and each tab is contained in exactly one of these cliques.*

Proof. First note that each of the four tabs are in exactly one such maximal 7-clique. Further, any other clique that contains three convex vertices has at least nine vertices: each convex vertex and its two reflex boundary neighbors. ⊠

We note that it is not necessary to identify the four tabs explicitly to continue with the reconstruction. There are only $7^4 = O(1)$ choices of tabs (one from each 7-clique of Lemma 4), thus we can try all possible tab assignments, continue with the reconstruction and verify that our reconstruction produces a valid IUP P with the same visibility graph. However, we can explicitly find the four tabs, giving us the following lemma.

Lemma 5. *We can identify the four tabs of an IUP in $O(nm)$ time.*

We pick one tab arbitrarily to be the north tab. We conceptually orient the polygon so that the northwest staircase is short and the northeast staircase is long. We do this by computing *elementary cliques*, which identify the convex vertices on the short staircase.

Definition 1 (elementary clique). *An elementary clique in an IUP is a maximal clique that contains exactly three convex vertices: one from a short staircase, and one from each of the long staircases. (See Fig. 2(a).)*

Lemma 6. *We can identify the elementary cliques containing vertices on the northwest staircase in $O(nm)$ time.*

Proof (Sketch). Each elementary clique is constant size and contains a 1-simplicial edge, and can therefore be discovered in $O(nm)$ time. Further, elementary cliques "interlock" along a staircase: each elementary clique shares exactly three reflex vertices with its at most two neighboring elementary cliques. Thus they can be computed starting from the elementary clique containing the north tab. ⊠

Note that, if our sole purpose is to reconstruct the IUP P, we have sufficient information. The number of elementary cliques gives us the number of vertices on a short staircase of the polygon, from which we can build a polygon. However, in what follows, we can actually map all vertices to their positions in the IUP, which we later use to build a recognition algorithm for IUPs.

First, we show how to assign all convex vertices from the elementary cliques to each of the three staircases, using visibility of the north and west tab vertices. Note, constructing the elementary cliques with Lemma 6 also gives us the west tab, since it is contained in the last elementary clique on the northwest staircase.

Lemma 7. *We can identify the convex vertices on the northwest staircase in $O(n)$ time.*

Proof. The northwest staircase contains the convex vertices of the elementary cliques from Lemma 6 that cannot be seen by any of the north or west tab vertices. The staircase further contains the left vertex of the north tab and the top vertex of the west tab (which can be identified by the fact that they are tab vertices that do not see either vertex of the other tab). ⊠

We can repeat the above process to identify the convex vertices of the southeast staircase. However, we might not yet be able to identify tabs as south or east. Thus, we will obtain two possible orderings of the convex vertices on the southeast staircase. Next, we show how to assign convex vertices to the long staircases. In the process we determine south and east tabs, and consequently, identify the correct ordering of convex vertices on the southeast staircase.

Lemma 8. *We can assign the remaining convex vertices in $O(n^2)$ time.*

Proof (Sketch). Let v_i, v_{i+1} be convex vertices on the same staircase, separated by a single reflex vertex. Let u be the unique vertex on the opposite staircase, such that the angular bisector of v_i goes through u. Then u sees v_{i+1}. This likewise holds for the opposite staircase. Therefore, starting from one convex vertex on each staircase (such as a tab vertex), we can compute all convex vertices on each staircase. ⊠
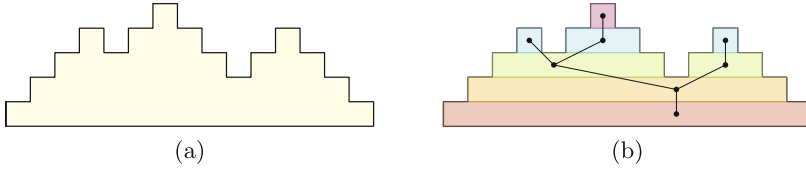
Fig. 3. (a) A histogram A with three tabs. (b) A decomposition of A into touching rectangles with a contact graph that is a tree.

Lemma 9. *We can assign the reflex vertices to each staircase in $O(n^2)$ time.*

Proof (Sketch). First we compare the reflex vertices seen by tab vertices, which gives us many vertices on the long staircases (see Fig. 2(b)). The remaining reflex vertices are discovered by building vertical and horizontal rectangles that contain unassigned reflex vertices (see Fig. 2(c)). ⊠

Observe that within each staircase, boundary edges are formed only between convex vertices and their reflex neighbors. Thus, we can order reflex vertices on each staircase by iterating over the staircase's convex vertices (order of which is determined in Lemmas 6, 7 and 8) and we are done. This gives us the following result:

Theorem 1. *In $O(n^2 m)$ time, we can reconstruct an IUP from its visibility graph.*

4 Uniform-Length Histogram Polygons

In this section we show how to reconstruct a more general class of uniform step length polygons: those that consist of a chain of alternating up- and down-staircases with uniform step length, which are monotone with respect to a single (longer) *base edge*. Such polygons are uniform-length *histogram polygons* [10], but we simply call them *histograms* for brevity (see Fig. 3(a) for an example). We refer to the two convex vertices comprising the base edge as *base vertices*. Furthermore, we refer to top horizontal boundary edges incident to two convex vertices as *tab edges* or just *tabs* and their incident vertices as *tab vertices*.

The Case of Two Staircases. We first note that in *double staircase* polygons (consisting of only two staircases) there is a simple linear-time reconstruction algorithm based on the degrees of vertices in the visibility graph. However, the construction relies on the symmetry of the two staircases and it is not clear whether any counting strategy works for arbitrary histograms.

4.1 Overview of the Algorithm

Every histogram can be decomposed into axis-aligned rectangles, whose contact graph is an ordered tree [10], as illustrated in Fig. 3(b). In Sect. 4.2, we show
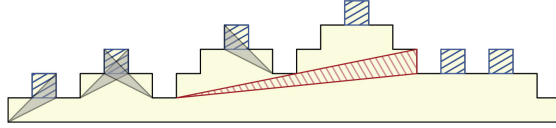
Fig. 4. Illustrating all maximal 4-cliques that contain 1-simplicial edges. These include tab cliques (square regions) and non-tab cliques (triangular regions).

that we can construct the (unordered) contact tree T from the visibility graph G_P in $O(n^2 m)$ time by repeatedly "peeling" tabs from the histogram. We then show that each left-to-right ordering of T's k leaves (as well as a left-to-right orientation of the rectangles in the leaves) induces a histogram P'. For each candidate polygon P' (of $k!2^k$ candidates), we then compute its visibility graph $G_{P'}$ in $O(n \log n + m)$ time [16] and check if $G_{P'}$ is isomorphic to G_P. Instead of requiring an expensive graph isomorphism check [6], we show how to use the ordering of T to quickly test if G_P and $G_{P'}$ are isomorphic.

In Sect. 4.4 we show how to reduce the number of candidate histograms from $k!2^k$ to $(k-2)!2^{k-2}$, leading to the main result of our paper:

Theorem 2. *Given a visibility graph G_P of a histogram P with $k \geq 2$ tabs, we can reconstruct P in $O(n^2 m + (k-2)!2^{k-2}(n \log n + m))$ time.*

Finally, we give a faster reconstruction algorithm when the histogram has a binary contact tree, solving these instances in $O(n^2 m)$ time (Sect. 4.4).

4.2   Rectangular Decomposition and Contact Tree Construction

We construct the contact tree T from G_P by computing a set $\mathcal{T}$ of the k tab edges of G_P (Lemma 11). Each tab (u,v) is 1-simplicial and in a maximal 4-clique, since $N(u) \cap N(v)$ is a 4-clique representing a unit square at the top of the histogram. Given the set $\mathcal{T}$ of tab edges, our reconstruction algorithm picks an edge t from $\mathcal{T}$ and removes the maximal 4-clique containing t. This is equivalent to removing an axis-aligned rectangle in P, and, equivalently, removing a leaf node from T. Moreover, it associates that node of T with four vertices of P: two *top vertices* that are convex and two *bottom vertices* that are either both reflex or are both convex *base* vertices. This process might result in a new tab edge, which we identify and add to $\mathcal{T}$.

Finding Initial Tabs. We start by finding the k tabs. Recall that every tab edge is 1-simplicial and in a maximal 4-clique. The converse is not necessarily true. Therefore, we begin by finding all 1-simplicial edges that are in maximal 4-cliques as a set of candidate edges, and later exclude non-tabs from the candidates.

Given a visibility graph $G_P = (V_P, E_P)$ of a histogram P and a maximal clique $C \subseteq V_P$, we call a vertex $w \in C$ an *isolated vertex with respect to P* if there exists a tab edge $(u,v) \in E_P$, such that $(N(u) \cup N(v)) \cap C = \{w\}$, i.e., of all vertices of C, only w is visible to some tab of P.
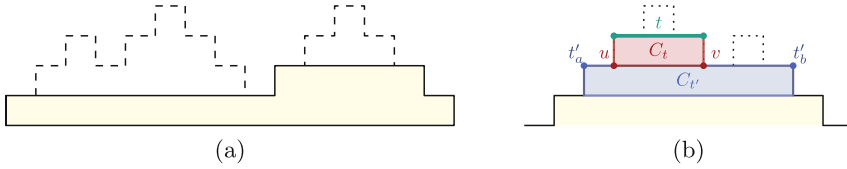
Fig. 5. (a) A truncated histogram, created by iteratively removing six tabs (dashed) from a histogram. (b) When removing C_t: t'_a, t'_b form a tab iff $t'_a, t'_b \in T \setminus C_t$, they see u and v, and $|C_{t'}| = 4$.

Lemma 10. *In a histogram, every 1-simplicial edge in a maximal 4-clique contains either a tab vertex or an isolated vertex.*

Proof (Sketch). Figure 4 shows the only types of maximal 4-cliques.   ⊠

Lemma 11. *In a visibility graph of a histogram, tabs can be computed in time $O(n^2m)$.*

Proof (Sketch). We find all maximal 4-cliques in $O(nm)$ by Observation 1 and detect and eliminate those containing isolated vertices in $O(n^2m)$ time.   ⊠

Note that top vertices cannot see the vertices above them. Therefore, only bottom vertices see tab vertices. Moreover, every bottom vertex sees at least one tab vertex. Thus, identifying all tabs immediately classifies vertices of G_P into top vertices and bottom vertices.

Peeling Tabs. Let P' be a polygon resulting from peeling tab cliques (rectangles) from a histogram P. We call P' a *truncated* histogram. See Fig. 5(a) for an example. After peeling a tab clique, the resulting polygon does not have uniform step length and the visibility graph may no longer have the properties on which Lemma 11 relied to detect initial tabs. Instead, we use the following lemma to detect newly created tabs during tab peeling.

Lemma 12. *When removing a tab clique from the visibility graph of a truncated histogram, any newly introduced tab can be computed in time $O(n)$.*

Proof. Denote the removed (tab) clique by C_t and let t be its tab. Let $u, v \notin t$ be the non-tab vertices of C_t. Since u sees v, (u, v) is an edge in G_P.

Since top vertices can only see vertices at and below their own level, besides the vertices of t, there are exactly two other top vertices in (remaining) G_P that see u and v, namely, the top vertices t'_a and t'_b of P on the same level as u and v (see Fig. 5(b)). Since t'_a, t'_b are adjacent in G_P, let $t' = (t'_a, t'_b)$.

When removing C_t from G_P, we can compute t'_a and t'_b in time $O(n)$ by selecting the only two top vertices adjacent to both u and v. Since t'_a and t'_b are the top vertices of a same rectangle $R_{t'}$, edge t' is 1-simplicial and is in exactly one maximal clique $C_{t'} = N(t'_a) \cap N(t'_b)$, which corresponds to the convex region $R_{t'}$. Finally, after C_t is removed, t' is a newly created tab if and only if $|C_{t'}| = 4$, which can again be tested in time $O(n)$ by computing $N(t'_a) \cap N(t'_b)$.   ⊠

With each tab clique (rectangle) removal, we iteratively build the parent-child relationship between the rectangles in the contact tree T as follows. Using an array A, we maintain references to cliques being removed whose parents in T have not been identified yet. When a tab clique C_t is removed from G_P, the reference to C_t is inserted into $A[u]$, where u is one of the rectangle's bottom vertices. If the removal of C_t creates a new tab $t' = (t'_a, t'_b)$, we identify $C_{t'}$ in $O(n)$ time using Lemma 12. Recall that t' sees all bottom vertices on the same level. Thus, for every bottom vertex $u \in N(t'_a)$ (in the original graph G_P), if $A[u]$ is non-empty, we set $C_{t'}$ as the parent of the clique stored in $A[u]$ and clear $A[u]$. This takes at most $O(n)$ time for each peeling of a clique. We get the following lemma, where the time is dominated by the computation of the initial tabs:

Lemma 13. *In $O(n^2 m)$ time we can construct the contact tree T of P, associate with each $v \in T$ the four vertices that define the rectangular region of v, and classify vertices of G_P as top vertices and bottom vertices.*

4.3   Mapping Candidate Polygon Vertices to the Visibility Graph

Let $\hat{T}$ correspond to T with some left-to-right ordering of its leaves and let $\hat{P}$ be the polygon corresponding to $\hat{T}$. We will map the vertices of G_P to the vertices of $\hat{P}$ by providing for each vertex of G_P the x- and y-coordinates of a corresponding vertex of $\hat{P}$. Let $t_1, t_2, \ldots, t_k$ be the order of the tabs in $\hat{P}$. Since $\hat{T}$ unambiguously defines the polygon $\hat{P}$, each node v of $\hat{T}$ is associated with a rectangular region on the plane, and the four vertices of G_P are associated with the four corners of the rectangular region. Since by Lemma 13 every vertex of G_P is classified as a top vertex or a bottom vertex, the y-coordinate can be assigned to all vertices unambiguously, because there are two top vertices and two bottom vertices associated with each node v of $\hat{T}$. For every pair $p, \bar{p}$ of top vertices or bottom vertices associated with a node in $\hat{T}$ (we call them *companion* vertices) there is a choice of two x-coordinates: one associated with the left boundary and one associated with the right boundary of the rectangular region. Thus, determining the assignment of each top vertex and bottom vertex in G_P to the left or the right boundary is equivalent to defining x-coordinates for all vertices in G_P. Although there appears to be $2^{n/2}$ possible such assignments, there are many dependencies between the assignments due to the visibility edges in G_P. In fact, we will show that by choosing the x-coordinates of the tab vertices, we can assign all the other vertices. Thus, in what follows we consider each of the 2^k possible assignments of x-coordinates to the $2k$ tab vertices.

At times we must reason about the assignment of a vertex to the left (right) staircases associated with some tab t_j. Given $\hat{T}$, the x-coordinates of each vertex in the left and right staircase associated with every tab t_j is well-defined. Therefore, assigning a vertex p to a left (right) staircase of some tab t_j defines the x-coordinate of p.

In a valid histogram, companion vertices p and $\bar{p}$ must be assigned distinct x-coordinates. Therefore, after each assignment below, we check the companion
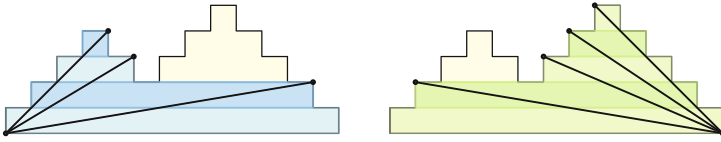
Fig. 6. Visibility from the left (right) base vertex determines the left- (right-)most tab, and orients all rectangles on the left (right) spine of the contact tree.

vertex and if they are both assigned the same x-coordinate, we exclude the current polygon candidate $\hat{P}$ from further consideration.

We further observe that in a valid histogram, if a bottom vertex p is not in the tab clique, then it sees exactly one tab vertex, which lies on the opposite staircase associated with that tab. Thus, we assign every such bottom vertex the left (right) x-coordinate if it sees the right (left) tab vertex.

Next, consider any node v of the contact tree $\hat{T}$ and let R_v define the rectangle associated with v in the rectangular decomposition of a valid histogram. Let p be a top vertex in R_v and let $S(p)$ be the set of vertices visible from p that are not in R_v ($S(p)$ can be determined from the neighborhood of p in G_P). Observe that if p is assigned the left (right) x-coordinate, then every vertex in $S(p)$ is a bottom vertex to the right (left) of the rectangle R_v, none of them belongs to a tab clique (i.e., all of them are already assigned x-coordinates), and all of them are assigned a right (left) x-coordinate. Since the x- and y-coordinates of the boundaries of R_v are well-defined by $\hat{T}$ (regardless of vertex assignment), if $S(p)$ is non-empty, we check all of the above conditions and assign p an appropriate x-coordinate. If a condition is violated, then the current polygon candidate is invalid and we exclude it from further consideration.

Let p be one of the remaining top vertices without an assigned x-coordinate. If the companion $\bar{p}$ is assigned an x-coordinate, we assign p the other choice of the x-coordinate. Otherwise, both p and $\bar{p}$ see only the vertices inside their rectangle. In this case, the neighborhoods $N(p)$ and $N(\bar{p})$ are the same and we can assign p and $\bar{p}$ to the opposite staircases arbitrarily.

Thus, the only remaining vertices without assigned x-coordinates are bottom vertices in tab cliques. Let R be the rectangle defined by the tab and $S_{right}(p)$ (resp., $S_{left}(p)$) denote the set of vertices that p sees among the vertices to the right (resp., left) of R. Consider a companion pair p and $\bar{p}$ of bottom vertices that are in a tab clique. Observe that if p is on the left boundary, then $S_{right}(\bar{p}) \subseteq S_{right}(p)$ or $S_{left}(p) \subseteq S_{left}(\bar{p})$. Symmetrically, if $\bar{p}$ is on the left boundary then $S_{right}(p) \subseteq S_{right}(\bar{p})$ or $S_{left}(\bar{p}) \subseteq S_{left}(p)$. Thus, if $|S_{right}(\bar{p})| \neq |S_{right}(p)|$ and $|S_{left}(p)| \neq |S_{left}(\bar{p})|$, we can assign p and $\bar{p}$ appropriate x-coordinates. Otherwise, the neighborhoods $N(p)$ and $N(\bar{p})$ are the same, and we can assign p and $\bar{p}$ to the opposite boundaries arbitrarily.

4.4 Reducing the Number of Candidate Histograms

We can reduce the number of possible orderings of tabs and staircases by considering only those that meet certain visibility constraints on the vertices that form the corners of each rectangle. In particular, we say that two rectangles $R_1 \neq R_2$ in the decomposition are *orientation-fixed* if a bottom vertex v_{bot} from one can see a top vertex v_{top} of another. Then these rectangles must be oriented so that v_{bot} and v_{top} are on opposite staircases (an up-staircase and a down-staircase). Thus, fixing an orientation of one rectangle fixes the orientation of the other.

Note that every rectangle is orientation-fixed with some leaf rectangle (as its bottom vertex can see a tab vertex). Therefore, ordering (and orienting) the leaves induces an ordering/orientation of the tree. There are $O(k!2^k)$ such orderings (and orientations) for all leaf rectangles, where k is the number of tabs.

For double staircases, T is a path and the root rectangle is orientation-fixed with every other rectangle (a base vertex is seen by every top vertex). Hence, orienting the base rectangle determines the positions of the top vertices on the double staircase. Likewise, for the histogram, the spines of T are fixed:

Lemma 14. *The base rectangle of a histogram is orientation-fixed with all rectangles on the left and right spines of T.*

Moreover, the only tab vertices visible from a base vertex are incident to the left-most or right-most tab. Thus, we can identify the left-most and right-most tabs based on the neighborhood of the base vertices. Note that removing a base rectangle of the histogram produces one or more histograms. Then we can apply this logic recursively, leading to the following algorithm:

1. Fix the orientation of the base rectangle. This identifies the rectangles on the left and right spines of T and their orientations. (See Fig. 6.)
2. The remaining subtrees collectively contain the remaining rectangles, which still must be ordered and oriented. We recursively compute the ordering and orientation of the rectangles in these subtrees.

Note if we compute the left and right spines of T, we identify the first and last tabs, and the orientations of their tab edges. Thus, we have $(k-2)!2^{k-2}$ remaining orderings of T and orientations of the tab edges to check, as $k - 2$ tabs remain. This results in the overall reconstruction of a histogram with $k \geq 2$ tabs in $O(n^2m + (k-2)!2^{k-2}(n \log n + m))$ time, proving Theorem 2.

We now generalize the number of orderings to consider by defining a recurrence on the tree structure. Let $v \in T$, and define $C(v)$ be v's children in T and $d(v) = |C(v)|$. Then if we have a fixed orientation of v's corresponding rectangle, fixing the rectangles on the left-most and right-most paths from v limits the number of possible orderings/orientations of v's descendants to

$$F(v) \leq \begin{cases} 2^{d(v)-2} \prod_{u \in C(v)} F(u) & \text{if } d(v) > 1, \\ F(u) & \text{if } |C(v)| = 1, \text{ s.t. } C(v) = \{u\}, \\ 1 & \text{if } v \text{ is a leaf.} \end{cases}$$

Note that $F(root) = 1$ for a binary tree T. That is, the orientation of the base rectangle completely determines the histogram. Furthermore, we can find such an orientation by fixing the orientation of the base edge, determining the left- and right-most paths, ordering and orienting them to match the base edge, and then repeating this for each subtree whose root is oriented and ordered (but its children are not), which acts as a base rectangle for its subtree. This process can be done in time $O(n + m)$ by traversing T and orienting each rectangle exactly once by looking at its vertices neighbors in its base rectangle in T.

Theorem 3. *Histograms with a binary contact tree can be reconstructed in $O(n^2 m)$ time.*

5   From Reconstruction to Recognition

We note that all of our reconstruction algorithms assign each vertex to a specific position in the constructed polygon. Let such an algorithm be called a *vertex assignment reconstruction*. As a result, we get recognition algorithms for these visibility graphs as well: we run our reconstruction until it fails or completes successfully, verify that the resulting polygon has the same visibility graph in time $O(n \log n + m)$ time [16], and verify that it is a polygon of the given type in linear time. Thus, we conclude that our reconstruction algorithms imply recognition algorithms with the same running times.

References

1. Abello, J., Eğecioğlu, Ö.: Visibility graphs of staircase polygons with uniform step length. Int. J. Comput. Geom. Ap. **03**(01), 27–37 (1993)
2. Abello, J., Eğecioğlu, Ö., Kumar, K.: Visibility graphs of staircase polygons and the weak Bruhat order, I: from visibility graphs to maximal chains. Discrete Comput. Geom. **14**(3), 331–358 (1995)
3. Abello, J., Kumar, K.: Visibility graphs of 2-spiral polygons (Extended abstract). In: Baeza-Yates, R., Goles, E., Poblete, P.V. (eds.) LATIN 1995. LNCS, vol. 911, pp. 1–15. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-59175-3_77
4. Abello, J., Kumar, K.: Visibility graphs and oriented matroids. Discrete Comput. Geom. **28**(4), 449–465 (2002)
5. Asano, T., Ghosh, S.K., Shermer, T.C.: Visibility in the plane. In: Urrutia, J.-R.S.J. (ed.) Handbook of Computational Geometry, pp. 829–876. North-Holland, Amsterdam (2000). Chap. 19
6. Babai, L.: Graph isomorphism in quasipolynomial time [extended abstract]. In: Proceedings of 48th ACM Symposium on Theory of Computing (STOC 2016), pp. 684–697, New York, NY, USA. ACM (2016)
7. Choi, S.-H., Shin, S.Y., Chwa, K.-Y.: Characterizing and recognizing the visibility graph of a funnel-shaped polygon. Algorithmica **14**(1), 27–51 (1995)
8. Colley, P.: Visibility graphs of uni-monotone polygons. Master's thesis, Department of Computer Science, University of Waterloo, Waterloo, Canada (1991)
9. Colley, P.: Recognizing visibility graphs of unimonotone polygons. In: Proceedings of 4th Canadian Conference on Computational Geometry, pp. 29–34 (1992)

10. Durocher, S., Mehrabi, S.: Computing partitions of rectilinear polygons with minimum stabbing number. In: Gudmundsson, J., Mestre, J., Viglas, T. (eds.) COCOON 2012. LNCS, vol. 7434, pp. 228–239. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32241-9_20
11. ElGindy, H.: Hierarchical decomposition of polygons with applications. Ph.D. thesis, McGill University, Montreal, Canada (1985)
12. Evans, W., Saeedi, N.: On characterizing terrain visibility graphs. J. Comput. Geom. **6**(1), 108–141 (2015)
13. Everett, H.: Visibility graph recognition. Ph.D. thesis, University of Toronto (1990)
14. Everett, H., Corneil, D.: Recognizing visibility graphs of spiral polygons. J. Algorithms **11**(1), 1–26 (1990)
15. Ghosh, S.K.: Visibility Algorithms in the Plane. Cambridge University Press, New York (2007)
16. Ghosh, S.K., Mount, D.M.: An output-sensitive algorithm for computing visibility. SIAM J. Comput. **20**(5), 888–910 (1991)
17. Jackson, L., Wismath, S.: Orthogonal polygon reconstruction from stabbing information. Comp. Geom. Theor. Appl. **23**(1), 69–83 (2002)
18. O'Rourke, J.: Art Gallery Theorems and Algorithms. Oxford University Press, New York (1987)
19. Sitchinava, N., Strash, D.: Reconstructing generalized staircase polygons with uniform step length. arXiv preprint https://arxiv.org/abs/1708.09842 (2017)