Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 26, no. 1, pp. 15–34 (2022)
DOI: 10.7155/jgaa.00579

Visibility Graphs of Anchor Polygons

Hossein Boomari[1] *Alireza Zarei*[1]

[1]Department of Mathematical Sciences,
Sharif University of Technology, Tehran, Iran

Submitted:	August 2016	Reviewed:	January 2021	Revised:	March 2021
Accepted:	November 2021	Final:	December 2021	Published:	January 2022
Article type:	Regular paper	Communicated by:	S. Whitesides		

Abstract. The visibility graph of a polygon corresponds to its internal diagonals and boundary edges. For each vertex on the boundary of the polygon, we have a vertex in this graph and if two vertices of the polygon see each other there is an edge between their corresponding vertices in the graph. Two vertices of a polygon see each other if and only if their connecting line segment completely lies inside the polygon. Recognizing visibility graphs is the problem of deciding whether there is a simple polygon whose visibility graph is isomorphic to a given graph. Another important problem is to reconstruct such a polygon if there is any. These problems are well known and well-studied, but yet open problems in geometric graphs and computational geometry. However, they have been solved efficiently for special cases where the target polygon is known to be a tower or a spiral polygon. In this paper, we propose a linear time algorithm to solve these recognizing and reconstruction problems for another type of polygons, named *anchor polygons*.

1 Introduction

The visibility graph, $G(V, E)$, of a simple planar polygon is a graph in which there is a vertex for each vertex of the polygon and for each pair of visible vertices of the polygon there is an edge between their corresponding vertices in this graph. Two points in a simple polygon are visible from each other if and only if their connecting segment completely lies inside the polygon. In this definition, each pair of adjacent vertices on the boundary of the polygon are assumed to be visible from each other. This implies that we have always a Hamiltonian cycle in a visibility graph which determines the order of vertices on the boundary of the corresponding polygon. We use the term "see" to address the visibility throughout this paper.

E-mail addresses: h.boomari1@sharif.edu (Hossein Boomari) zarei@sharif.edu (Alireza Zarei)

1.1 Related Works

Computing the visibility graph of a given simple polygon has many applications in computer graphics [14], computational geometry [11] and robotics [4]. There are several efficient polynomial time algorithms for this problem [11].

This concept has been studied in reverse as well: Is there any simple polygon whose visibility graph is isomorphic to a given graph and if there is such a polygon, is there any way to reconstruct it (finding positions for its vertices on the plain)? The former problem is known as recognizing visibility graphs and the latter one is known as reconstructing polygon from visibility graph. Both these problems are widely open. The only known result about the computational complexity of these problems is that they belong to *PSPACE* [7] complexity class and more specifically belong to the class of *existential theory of the reals* [12]. This means that it is not even known whether these problems are *NP-Complete* or can be solved in polynomial time. Even if we are given the Hamiltonian cycle of the visibility graph, which determines the order of vertices on the boundary of the target polygon, the exact complexity class of these problems is still unknown.

However, these problems have been solved efficiently for special cases of tower and spiral polygons. In these special cases, we know that the given pair of the graph and the Hamiltonian cycle corresponds to a tower or a spiral polygon. A tower polygon consists of two concave chains on its boundary who share one vertex, and the other end points are connected by a segment (see Figure 1.a). A spiral polygon has exactly one concave and one convex chain on its boundary (see Figure 1.b). The recognizing and reconstruction problems have been solved for tower polygons in linear time in terms of the size of the graph [5]. It has been shown in [5] that a given graph is the visibility graph of a tower polygon if and only if after removing the edges of the Hamiltonian cycle from the graph, an isolated vertex and a connected bipartite graph are obtained and the bipartite graph has *strong ordering* following the order of vertices in the Hamiltonian cycle. A strong ordering on a bipartite graph $G(V, E)$ with partitions U and W is a pair of $<_V$ and $<_W$ orderings on respectively U and W such that if $u <_U u'$, $w <_W w'$, and there are edges (u, w') and (u', w) in E, the edges (u', w') and (u, w) also exist in E. Graphs with strong ordering are also called *strong permutation graphs*. The recognizing and reconstruction problems have also been solved efficiently for spiral polygons [8]. Since we need this method in our algorithm, we describe it in more details in Section 2.

Although there is a bit progress on recognizing and reconstruction problems, there have been plenty of studies on characterizing visibility graphs. In 1988, Ghosh introduced three necessary conditions for visibility graphs and conjectured their sufficiency [9]. These conditions are mainly based on the definition of *blocking vertices*. A *blocking vertex* for a non-visible pair $\langle a, b\rangle$ refers to a vertex p, that its existence blocks the visibility of the vertices between a and p in the boundary of the polygon to all of the vertices between p and b. In 1990, Everett proposed a graph that rejects Ghosh's conjecture [7]. He also refined Ghosh's third necessary condition to a new stronger one [10]. In 1992, Abello *et al.* built a graph satisfying Ghosh's conditions and the stronger version of the third condition which was not the visibility graph of any simple polygon [3] disproving the sufficiency of these conditions. In 1997, Ghosh added his forth necessary condition and conjectured that this condition along with his first two conditions and the stronger version of the third condition are sufficient for a graph to be a visibility graph. Finally, in 2005 Streinu proposed a counterexample for this conjecture [13]. Independently in 1994, Abello *et al.* proposed the notion of *q-persistant* graphs, that includes visibility graph and *blocking vertex assignment*. A *Blocking vertex assigment* is a proper function from non-visible ordered pairs to a blocking vertex that satifies four conditions. In this definition, blocking vertex assigment assigns each non-visible ordered pair $\langle a, b\rangle$ to one of its blocking vertices which is visible from a. They proved that each visibility graph has at least one blocking vertex assigment. They conjectured that these constraints are verifiable efficiently [1]. But, the computational complexity of verifying the existence of a blocking vertex assigment or finding such a function in a visibility graph, are not known to be solvable in polynomial time. Later in 1995, Abello *et al.* showed that, these constraints, along with another constraint are sufficient for recognizing and reconstruction of 2-spiral polygons[1], i.e. given a graph, its Hamiltonian cycle and a blocking vertex assigment, they solved the recognizing and reconstruction problems for 2-spiral polygons, efficiently [2]. But by now, there is no efficient algorithm for obtaining a blocking vertex assignment for 2-spiral polygons from which the recognizing and reconstruction problems could be solved efficiently for this type of polygons.

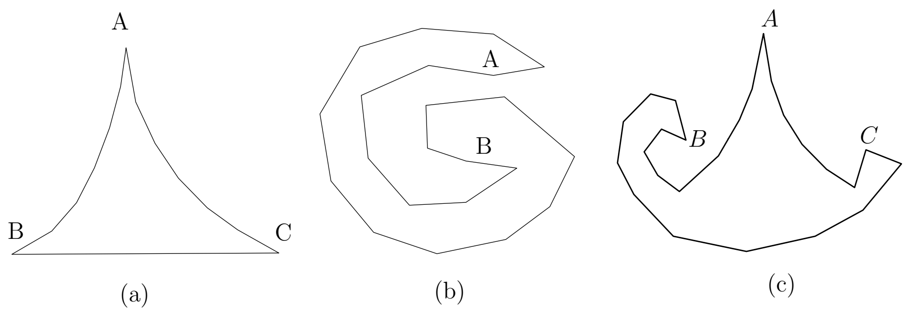

Figure 1: a) Tower polygon, b) Spiral polygon, c) Anchor polygon.

1.2 Our Result

In this paper, we consider these problems for another type of polygons called *anchor polygons*. The boundary of an anchor polygon is composed of two concave chains and a convex one (see Figure 1.c). We characterize these polygons with a set of efficiently realizable constraints and show that both recognizing and reconstruction problems for a given pair of visibility graph and Hamiltonian cycle belonging to an anchor polygon can be solved in $O(|E|)$ time or equivalently, linear in terms of the size of the input.

In the remainder of this paper, we first introduce the algorithm of solving reconstruction and recognizing problems for spiral polygons in Section 2. Also, in this section we present some preliminaries and definitions used in next sections. In Section 3, we give an overview of our method and extract key features from the graph, used in our reconstruction algorithm. In Section 4 we present our recognizing and reconstruction algorithms. We analyze their efficiency in Section 5.

2 Preliminaries and Definitions

We first briefly describe the recognizing and reconstruction algorithm for spiral polygons in Section 2.1. We need these details in some parts of our algorithm. Then, we introduce required notations and definitions in Section 2.2, and basic facts and properties in Section 2.3 used in the rest of the paper.

[1]polygons with at most 2 concave chains

2.1 Spiral Polygons

Assume that a pair of visibility graph and Hamiltonian cycle is given. Everret and Corneil proposed an efficient method to test whether such a pair belongs to a spiral polygon and reconstruct it if the test passed [8]. Here, we briefly describe their method.

The visibility graph of a spiral polygon is a limited subclass of *interval graphs* [8]. This means that any interval graph satisfying a certain necessary condition corresponds to the visibility graph of a spiral polygon and vice versa (For the sake of brevity we skip describing this extra condition).

In such an interval graph, and equivalently in the visibility graph of a spiral polygon, there are at least two vertices which form cliques with all their neighbors, called *joint vertices*. Moreover, by removing one of these vertices, the remaining graph is still an interval graph. In a spiral polygon the joint vertices that connect the convex and concave chains have this property (form a clique with their neighbors) and by removing one of these vertices the residual graph will be another spiral polygon. Performing this *elimination scheme* from one of the joint vertices toward the other one will finally give us an ordered sequence of removed vertices and a subset of vertices which is a clique composed of the other joint vertex and its neighbors. During this process the Hamiltonian cycle H corresponding to the boundary of the realized spiral polygon, is computed as well. Denote the vertices of the convex and concave chains as *convex* and *concave* vertices, respectively. Note that a concave vertex blocks the visibility of its adjacent vertices. Therefore, a vertex whose adjacent vertices see each other must be a convex vertex.

Assume that $< v_1, v_2, ..., v_k >$ is the ordered sequence of the removed vertices in the above procedure and $\{v_{k+1}, ..., v_n\}$ is the set of remaining vertices, which is a clique. Assume that v_n is the joint vertex and v_{k+1} is the only concave vertex in this set. There is only one concave vertex in this clique (which is the one adjacent to v_n) because two non-adjacent vertices on a concave chain are non-visible and do not exist in the clique. The visibility graphs of all convex polygons with the same number of vertices are isomorphic, i.e. they are complete graphs. Therefore, we put $\{v_{k+1}, ..., v_n\}$ on the boundary of an arbitrary convex polygon with respect to their order in H (see Figure 2). Then, the position of the vertices $< v_1, v_2, ..., v_k >$ are located in reverse order inductively as follows: For an arbitrary vertex v_l in $< v_1, v_2, ..., v_k >$, assume that v_c is the last located convex vertex before v_l, and v_r is the last located concave vertex before v_l. For the induction base step, we set $v_l = v_{k+1}$, $v_r = v_n$ and $v_c = v_{k+2}$. To locate the position of v_{l-1} in an inductive step, assume that v_t is the closest convex vertex to v_r which sees v_{l-1}. By closest we mean that v_t is the first vertex on the Hamiltonian cycle when we move from v_r along the reconstructed part of the concave chain and continue along the reconstructed part of the convex chain (see arrows in Figure 2). If v_l is a convex vertex, v_{l-1} is located somewhere inside the angle $\widehat{v'_t v_r v'_{t-1}}$ (see Figure 2.a) where $v_r v'_t$ (resp. $v_r v'_{t-1}$) is the half line from v_l along $v_r v_t$ (resp. $v_r v_{t-1}$) and in opposite side of v_t (resp. v_{t-1}) and in such a way that the new boundary does not cross itself. To add this vertex to the constructed polygon we remove the edge between v_r and v_l and add two new edges connecting v_r to v_{l-1} and v_l to v_{l-1}. Otherwise (if v_l is a concave vertex), v_{l-1} is located somewhere inside angle $\widehat{v'_t v_l v'_{t-1}}$ (see Figure 2.b) where $v_l v_t$ and $v_l v_{t-1}$ half-lines are defined similarly. This is done by removing the edge $v_c v_l$ and adding edges $v_l v_{l-1}$ and $v_c v_{l-1}$.

An important feature of this reconstruction algorithm is that starting from the initial convex polygon $v_{k+1}, ..., v_n$, the remainder of the spiral polygon can be reconstructed in an arbitrary small area close to the concave vertex of this convex polygon. We use this feature in our reconstruction algorithm.

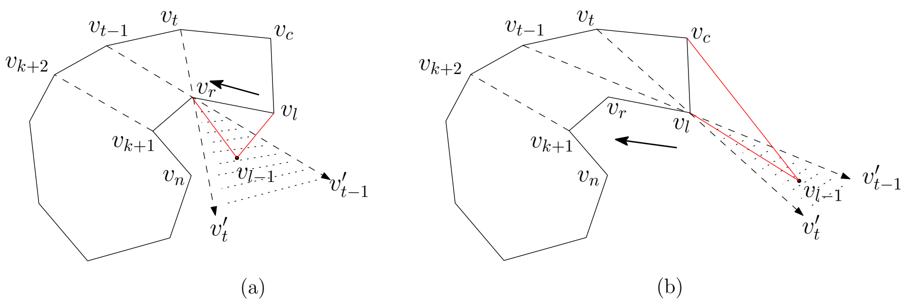

Figure 2: The reconstruction algorithm for spiral polygons

2.2 Definitions

In an anchor polygon, there are three specific vertices joining the three chains on the boundary of such a polygon. As shown in Figure 1.c, the joint vertex between the concave chains is named A and the other joint vertices are named B and C. Without loss of generality, we assume that we have a left concave chain from A to B and a right concave chain from A to C and an underneath convex chain from B to C. We may refer to A as top joint vertex and to B and C as the left and right joint vertices, respectively. These names are consistent in all figures and inside the text to help readers having better perspective about the target anchor polygon.

We denote by pq^i the i^{th} vertex on the boundary of the polygon when we move from vertex p to vertex q, which both lies on the same chain. For example, AB^0 is the joint vertex A and AB^1 is the first vertex after A on the left concave chain. We also use $xy(p)$ as the closest vertex to x on chain xy which is visible to vertex p. In this notation, x and y may be any of the joint vertices A, B or C. Figure 3 illustrates the definitions.

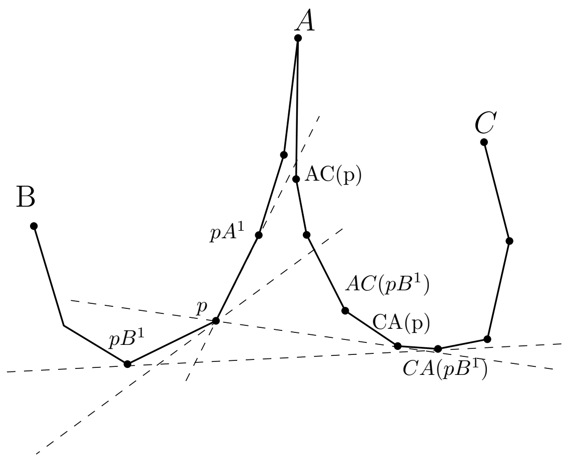

Figure 3: Illustrating the definitions

2.3 Basic Facts

From the convexity or concavity of the chains we have the following basic observations. Note that the first two observations are only based on the properties of convex and concave vertices, so are valid on spiral polygons as well.

Observation 1 *Adjacent vertices (in the given Hamiltonian cycle) of a vertex $a \notin \{A, B, C\}$ of concave chains are non-visible.*

Observation 2 *Two vertices of the convex chain BC see each other if and only if no vertex of the concave chains blocks their visibility.*

Observation 3 *If a vertex v_i on the convex chain sees another vertex v_j on this chain, all vertices from v_i to v_j on this chain see each other. Moreover, if a vertex v_i on the convex chain does not see another vertex v_k on this chain and v_i is closer to B on chain BC, then none of the vertices from B to v_i see any one of the vertices from v_k to C (see Figure 4).*

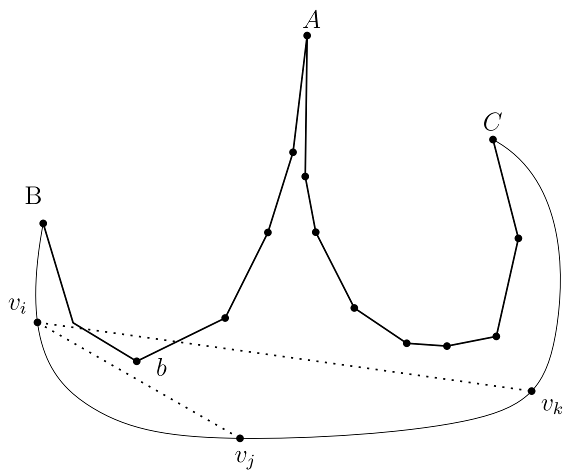

Figure 4: Visibility of the vertices on the convex chain

3 Determining Joint Vertices

We propose a constructive algorithm to solve both recognizing and reconstruction problems for anchor polygons. In this algorithm, we first determine the three chains on the given Hamiltonian cycle. During this process, some necessary conditions of the recognizing algorithm are verified. Then, the area of the target polygon is decomposed into four sub-polygons: a tower polygon, a convex polygon and two spiral polygons. The tower polygon is reconstructed first. Then, the convex polygon is constructed under the base edge of the built tower, and, finally the spiral polygons are built and attached to the sides of the constructed tower and convex polygon (see Figure 8). During this reconstruction process, the necessary conditions of the given visibility graph are checked to have a recognizing algorithm as well as the reconstruction one.

The details of the decomposition and reconstruction phases are described in Section 4. Here, we give a method for identifying the joint vertices A, B and C from which the three chains on the boundary of the target polygon are obtained. To do this, we first assume that we know the joint vertex A and propose an algorithm for finding vertices B and C. Then, we propose a method for identifying candidate vertices for A.

3.1 Finding joint vertices B and C

We begin this section with two observations:

Observation 4 *The vertices AB^1 and AC^1 are always visible from each other.*

Observation 5 *At least one of the pairs of vertices $\langle BC^1, BA^1\rangle$ and $\langle CB^1, CA^1\rangle$ are visible from each other (Figure5a provides an instance in which BC^1 and BA^1 are non-visible).*

Proof: The blocking vertices of $\langle BC^1, BA^1\rangle$ must lie on chain AC and blocking vertices of $\langle CB^1, CA^1\rangle$ must lie on chain AB. If BC^1 and BA^1 are non-visible, then none of the vertices of the chain AB can block the visibility of CB^1 and CA^1. □

When we move from A on the Hamiltonian cycle in both directions, observations 1 and 5 imply that we can find at least one of the joint vertices B or C. This is the first visited vertex in these walks whose adjacent vertices in the Hamiltonian cycle see each other. Our algorithm for finding the other vertex is exactly the same: Walk along the Hamiltonian cycle from A in both directions until a vertex with this property (its adjacent vertices in Hamiltonian cycle see each other) is found in each direction. This algorithm will successfully find correct vertices as B and C if both pairs $\langle BA^1, BC^1\rangle$ and $\langle CA^1, CB^1\rangle$ are visible from each other. But, in some cases one of these pairs are non-visible. Then, it seems that, our algorithm fails to find joint vertex B or C.

We assume that both concave chains have at least one vertex other than the joint vertices. Otherwise, the target polygon will be a spiral one and can be recognized and reconstructed by the algorithm proposed by Everett and Corneil [8]. Assume that $G(V, E)$ and H are the given pair of visibility graph and Hamiltonian cycle. The following theorem shows that if $G(V, E)$ and H belong to an anchor polygon, the joint vertices B and C (obtained from the above algorithm) along with A are the joint vertices of an anchor polygon whose visibility graph and Hamiltonian cycle are equivalent to the given pair of $G(V, E)$ and H.

Theorem 1 *Assume that for a given visibility graph $G(V, E)$ and Hamiltonian cycle H and a vertex A, the vertices B' and C' are the first visited vertices on H when we walk from A in both sides whose adjacent vertices see each other. Then, G and H correspond to an anchor polygon P with top vertex A, if and only if there is an anchor polygon P' with joint vertices A, B' and C' whose visibility graph and Hamiltonian cycle are respectively isomorphic to G and H.*

Proof: Trivially, if P' exists we can consider it as P as well which implies the theorem in one direction. For the other direction, if the joint vertices B and C of P are respectively equal to B' and C', then theorem is true by considering $P = P'$.

Therefore, it is enough to prove the theorem for the cases where G and H belong to an anchor polygon P with joint vertices A, B and C and either $B \neq B'$ or $C \neq C'$. For these cases, we must prove that there exists another anchor polygon P' with joint vertices A, B' and C' whose visibility graph and Hamiltonian cycle are respectively isomorphic to G and H. According to Observation 5 and without loss of generality, assume that $C = C'$ and $B \neq B'$. This means that B' lies on the convex chain of P, and equals to a vertex BC^i for some $i > 0$, while $B = BC^0$ in the anchor polygon P.

We complete the proof of the theorem by showing that if BA^1 and BC^1 do not see each other in P, we can consider the joint vertex B as a vertex of the left concave chain and BC^1 as the left joint vertex, and build an anchor polygon, whose visibility graph and Hamiltonian cycle are respectively isomorphic to G and H. This has been shown in Figure 6 where (a) is the original polygon and (b) is the new one with BC^1 as a joint vertex. Then we can repeat this process i times and the result is a new anchor polygons with joint vertices A, C and BC^i whose visibility graph and Hamiltonian cycle are respectively isomorphic to G and H.

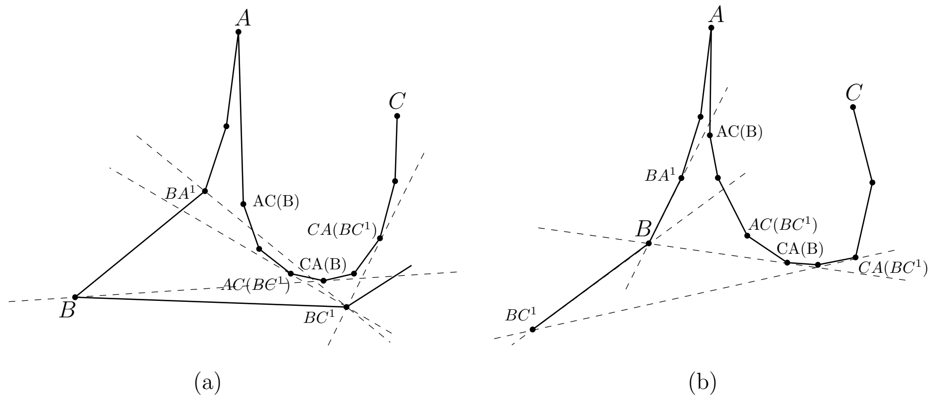

Figure 5: Considering BC^1 as a joint vertex in (b) when BC^1 and BA^1 are non-visible in (a).

Therefore, it is enough to show that it is always possible to locate BC^1 on the left concave chain of some anchor polygon without disturbing the visibility graph constraints. This is done in two steps. In the first step, we prove that the induced visibility graph on vertices $U = \langle A, AB^1, ..., B, BC^1\rangle$ and $W = \langle A, AC^1, ..., CA(BC^1)\rangle$ have strong ordering with partitions U and W and, then, we can build a tower polygon on these vertices in which B is a concave vertex on the left chain and BC^1 is the last vertex on this concave chain. In the second step we prove that it is possible to extend the concave chain W by adding vertices $\langle CA(BC^2), ..., C\rangle$ and locating the vertices $\langle BC^2, ..., C\rangle$ on a convex chain from BC^1 to C satisfying the visibility graph constraints.

For the first step, we know from P that the induced visibility graph on $U' = \langle A, AB^1, ..., B\rangle$ and $W' = \langle A, AC^1, ..., CA(B)\rangle$ have strong ordering. Then, it is enough to consider the pairs $\langle BC^1, w\rangle$ and $\langle u, w'\rangle$, where $w, w' \in W$ and $w <_W w'$, w is visible from BC^1, and $u \in U$ is visible from w' and prove that both (u, w) and (BC^1, w') exist in the visibility graph. Since w' is a vertex on chain AC between vertices w and $CA(BC^1)$, it must be visible from BC^1, which means that (BC^1, w') exists in the graph. On the other hand, all vertices of AC from w' to w are visible from B. Thus, if $u = B$, this means that (u, w) exists in the visibility graph and otherwise, if $u \neq B$, the visibility of u and w is derived from strong ordering on U' and W' (existence of (B, w) and (u, w') implies existence of (u, w)). This completes the existence of the strong ordering on U and W.

For the second step, we show that after building the tower polygon on U and W, we can add the remained vertices to build an anchor polygon with A, BC^1 and C as its joint vertices with the same visibility graph as P. These remaining vertices are the vertices of the right concave chain of

P from C to $CA(BA^1)$ and the vertices of convex chain from C to BC^1. We add the first set as a concave chain of a spiral polygon starting from $CA(BC^1)$ and above the line passing through vertices BC^1 and $CA(BC^1)$. This is consistent with the visibility graph because none of these vertices see any vertex of the tower (see Figure 5.b). However, these vertices must be located in such a way that support the visibility relations with vertices of the the convex chain. The convex vertices from C to $BC^1 = B'$ are divided into three parts:

- (V_1) The visible vertices from B (they are also visible from B')
- (V_2) The visible vertices from B' which are non-visible to B
- (V_3) The vertices which are non-visible from B'

It is simple to check that none of these vertices are visible from BA^1. According to Figure 6, assume that d_1 is the line passing through BA^1 and $CA(BA^1)$, d_2 is the line from B to $CA(B)$ and d_3 is the line from B' to $CA(B')$. The vertices of $V_1 \cup V_2 \cup V_3$ are put on a convex chain from B' to C in such a way that vertices of V_1, V_2 and V_3 lie inside α, β and γ, respectively. As the slope of d_3 is higher than the slope of d_2 and the slope of d_2 is higher than the slope of d_1, both α, β and γ are non-empty. Therefore, by considering these constraints, our placement supports visibility edges between vertices of the left chain and convex vertices. We must be careful only about the visibility edges between the right chain and the convex vertices. As shown in Figure 6, all of the vertices of the right chain from A to $CA(BA^1)$ except $CA(BA^1)$, are non-visible to the vertices of the convex chain. Hence, by placing convex vertices above d_1 (which includes both regions α and β) these non-visibilities are satisfied. On the other hand, all vertices of the right concave chain lie above the line through B' and $CA(B')$ which means that all visible vertices to B' from the convex chain ($V_1 \cup V_2$) are also visible to $CA(B')$. Therefore, for any convex vertex $p \in V_1 \cup V_2$, p sees all vertices of the right chain from $CA(B')$ to $CA(p)$. Moreover, $CA(p)$ blocks the visibility of p and all vertices of the right chain from A to $CA(p)$. For such a vertex p define the line through $CA(p)A^1$ and $CA(p)$ as d_p and the line through $CA(p)$ and $CA(p)C^1$ as d'_p. Hence, p must be placed somewhere above the line d_p and below the line, d'_p (see Figure. 6). For any $p' \in V_1$ (resp. $p' \in V_2$), which is closer to C than p, the slope of $d_{p'}$ is greater than slopes of d_p and d_1 (resp. d_2) and is less than the slope of d_2 (resp. d_3) and the slope of $d'_{p'}$ is higher than the slope of d'_p and all these lines intersect each other above d_3. Therefore, we can place the vertices of $V_1 \cup V_2$ on an arbitrary convex curve from B', which completely lies below d_3 and intersects d_1, d_2 and $d_{CB(B')}$ strictly below d_3, according to their order in the Hamiltonian cycle, in such a way that for each vertex $p \in V_1 \cup V_2$, p is located above the line d_p and below the line d'_p, while they support the visibility edges of the induced visibility graph on vertices of the sub-chain from A to $CA(B')$ and vertices of $V_1 \cup V_2$.

The convex vertices from B' to C and the concave vertices from $CA(B')$ to C build a spiral polygon and we have already built a convex sub-polygon of it with boundary vertices $\{B', CA(B')\} \cup V_1 \cup V_2$ (with B' as one of its joint vertices, and other vertices as neighbors of B'). The remaining vertices of this spiral polygon (equivalently, the remaining vertices of the anchor polygon) are reconstructed according to the method described in Section 2 in a close neighborhood of $CA(B')$ in such a way that its boundary does not intersect the built part of the right concave chain of the anchor polygon. Note that this reconstruction forces the vertices of V_3 to be located above d_3. This completes our proof, when considering BC^0 as a concave vertex and BC^1 as a joint vertex.

As said before, we can repeat this process i times to obtain an anchor polygon with A, BC^i and C joint vertices and G and H as its visibility graph and Hamiltonian cycle, respectively.

□

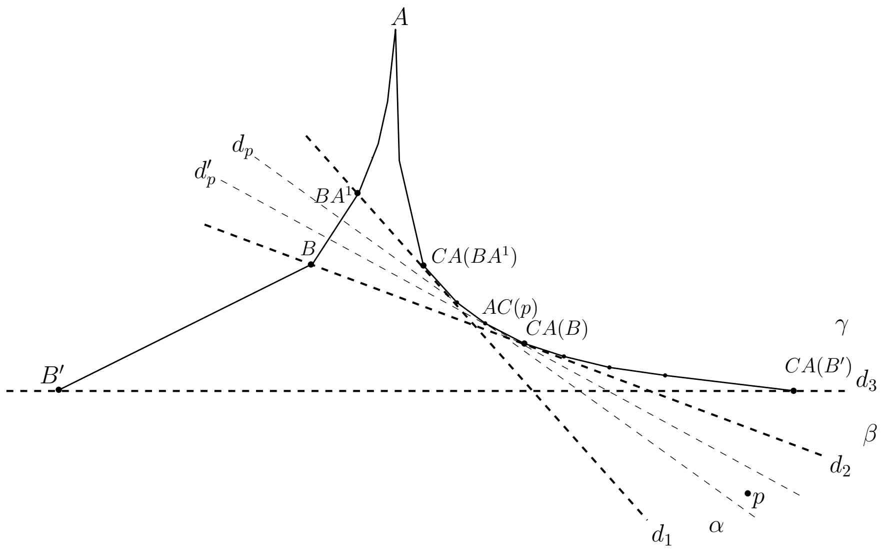

Figure 6: Lines d_1 and d_2, parts α and β and other constraints

3.2 Determining joint vertex A

As an important part of our recognition algorithm, we describe a method for identifying the joint vertex A. If one of the concave chains AB or AC has only one edge (two vertices), the target polygon will be a spiral one and recognizing and reconstruction problems can be solved in such cases using the method proposed by Everett and Corneil [8]. Therefore, we assume that both chains AB and AC have at least one non-joint vertex. Then, the following observation is true for the joint vertex A.

Observation 6 *The pairs $\langle A, AB^2\rangle$ and $\langle A, AC^2\rangle$ do not see each other.*

From this observation we have necessary conditions to find candidate vertices for A. We use these conditions in the first phase of our algorithm by moving along the Hamiltonian cycle and finding those vertices whose adjacent vertices see each other, but such that the two vertices at distance 2 in the Hamiltonian cycle do not see these vertices. Then, we use our algorithm for finding other joint vertices (B and C) corresponding to any one of the candidate vertices for A. Clearly, for any candidate vertex p for A we must find corresponding joint vertices B_p and C_p where chain B_pC_p is convex. Each pair of visible vertices in convex chain B_pC_p must satisfy Observation 3. We show that there are at most three candidate vertices for A which satisfy the above conditions.

Assume that the given pair of visibility graph and Hamiltonian cycle belongs to an anchor polygon P with joint vertices A_P, B_P and C_P.

Theorem 2 *If our algorithm finds another candidate vertex A' for the top joint vertex A other than A_P, B_P and C_P, then A' and its corresponding other joint vertices $B_{A'}$ and $C_{A'}$ must lie on the convex chain B_PC_P of P. This means that both chains A_PB_P and A_PC_P must lie on the convex chain $B_{A'}C_{A'}$ of the candidate top joint vertex A'.*

Proof: Assume that a vertex A' satisfies all conditions we check in our algorithm for finding candidate vertex A. By Observation 4, the adjacent vertices of A' must see each other and it must be located on the chain BC. On the other hand, our algorithm finds the other joint vertices $B_{A'}$ and $C_{A'}$ with respect to A', as the first vertices whose adjacent vertices are visible. In addition, both pairs adjacent to B_P and C_P are visible. Therefore, both vertices $B_{A'}$ and $C_{A'}$ lie on the convex chain B_PC_P of P which proves the theorem. In addition, both chains A_PB_P and A_PC_P must lie on the convex chain $B_{A'}C_{A'}$ of the candidate top joint vertex A'. □

Theorem 3 *Our algorithm finds at most one candidate vertex A' for the top joint vertex A out of $\{A_P, B_P, C_P\}$. Moreover, if any one of the joint vertices B_P and C_P is a candidate vertex for A, there can be no more candidate vertex on the convex chain B_PC_P out of B_P and C_P.*

Proof: For the sake of a contradiction, assume that our algorithm finds two candidate vertices A_1 and A_2 for the top joint vertex A out of $\{A_P, B_P, C_P\}$. By Theorem 2, both these vertices and their corresponding other joint vertices must lie on the convex chain B_PC_P in P. Without loss of generality, assume that A_2 lies between B_P and A_1 on this convex chain (see Figure 7). From the definition of joint vertices B and C and conditions for the top joint vertex, A_2 and $A_2{B_P}^2$ must not be visible and $B_P{A_P}^1$ and $B_P{C_P}^1$ must be visible pairs. This forces that there must be at least one vertex between B_P and A_2 which means that $A_2{B_P}^1$ cannot be equal to B_P. While A_2 and $A_2{B_P}^2$ are a non-visible pair on the convex chain of P, there must be a blocking vertex b on A_PB_P or A_PC_P chains and visible to A_2 preventing their visibility. Recall that our algorithm for finding joint vertices of a top joint vertex seeks for the first vertices whose adjacent vertices see each other. Both pairs of adjacent vertices to A_2 and C_P are visible. Therefore, both of the corresponding joint vertices of the top joint vertex A_1 (according to our algorithm) lie between vertices A_2 and C_P on the convex chain of P. This implies that all vertices of the chain B_PC_P from A_2 to B_P and vertices of the chains A_PB_P and A_PC_P in P lie on the convex chain of the candidate top joint vertex A_1. From Observation 3, when two vertices A_2 and b on this convex chain see each other, all vertices from A_2 to b, including $A_2{B_P}^2$, must also see each other and form a clique which is a contradiction.

By the same argument we can prove that if the joint vertex C_P(or B_P) is a candidate for the top joint vertex, there cannot be any other candidate vertex for the top joint vertex on the convex chain B_PC_P out of $\{B_P, C_P\}$. □

From the above theorems, we conclude that according to our algorithm there will be at most three candidates for the top joint vertex A. Precisely, if there was any other candidate other than A, either it is a vertex A' on B_PC_P ($A' \notin \{B_P, C_P\}$) or we have at most two candidates from B and C.

4 The Reconstruction Algorithm

In this section, we assume that we are given a pair of visibility graph, $G(V, E)$ and Hamiltonian cycle, H, and three joint vertices A, B and C and the goal is to obtain an anchor polygon $G(V, E)$ corresponding to these graph and cycle with A, B and C as its top, left and right joint vertices, respectively. Moreover, we assume that the visibility graph and the joint vertices satisfy conditions described in previous observations and conditions of previous algorithms (the joint vertices have been obtained by the algorithms described in Section 3). From the previous section, we know that there are at most three options for these joint vertices and the adjacent vertices to each of these joint vertices are visible. Therefore, to solve the recognizing algorithm we may run the following

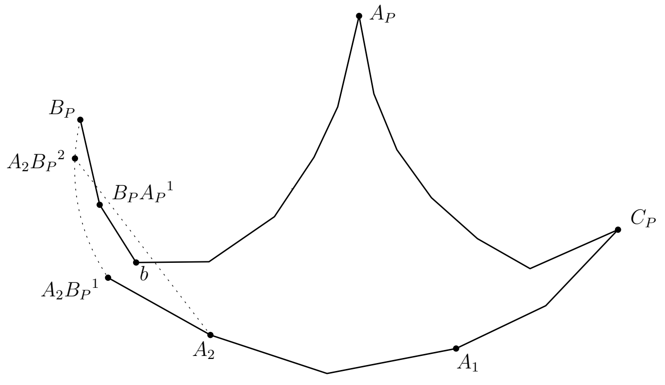

Figure 7: A_2 and $A_2{B_P}^2$ must not be visible and $A_2{B_P}^1$ and $A_2{C_P}^1$ must be visible pairs.

algorithm at most three times and if one of these runs leads to an anchor polygon it will be returned as a solution otherwise, if none of them produces a polygon it means that $G(V,E)$ and H do not belong to an anchor polygon.

Our reconstruction algorithm consists of two phases. Initially we decompose the target polygon into at most four regions and then these regions are reconstructed to build the final anchor polygon.

4.1 Anchor polygon decomposition

We define a line d as a bi-tangent line for chains AB and AC if it passes through vertices M and M' on AB and AC, respectively, and both chains lie completely on the same side of it (see Figure 8). The existance and position of this bi-tangent is identified by the following observation to simplify next references.

Observation 7 *Each anchor polygon has exactly one bi-tangent line and this bi-tangent line passes through M as the last vertex of chain AB, which is visible from some vertex of AC chain, and M' as the last vertex of chain AC, which is visible from some vertex of AB chain.*

According to the above observation, this bi-tangent line can be computed from the visibility graph as follows: Let M (resp. M') be the furthest vertex from A in AB (resp. AC) that sees a vertex of AC (resp. AB). Then, there is no edge from vertices of AM (resp. AM') to vertices of $M'C$ (resp. MB) except the edge MM'. Notice that the polygon with boundary $\langle M, ..., A, ..., M'\rangle$ is a tower polygon.

Let $N = BC(M')$ and $N' = CB(M)$ be respectively the first (starting respectively from B and C) visible vertices from M' and M on chain BC (see Figure 8). We find N (resp. N'), by simply searching for the furthest vertex from B (resp. C) in BC, which is visible to M' (resp. M). As we stated before, the polygon with boundary vertices $\langle M, ..., A, ..., M'\rangle$ is a tower polygon and the polygon with boundary vertices $\langle M, N, ..., N', M'\rangle$ is convex and both polygons with boundaries $\langle M, ..., B, ..., N\rangle$ and $\langle M', ..., C, ..., N'\rangle$ are spiral polygons (see Figure 8).

Observation 8 *The definition of vertices $\{M, M', N, N'\}$ forces that there is no edge between a vertex of any one of these spiral polygons to a vertex of the other spiral polygon or to a vertex of the tower sub-polygon, except edges with an end point in $\{N, N', M, M'\}$.*

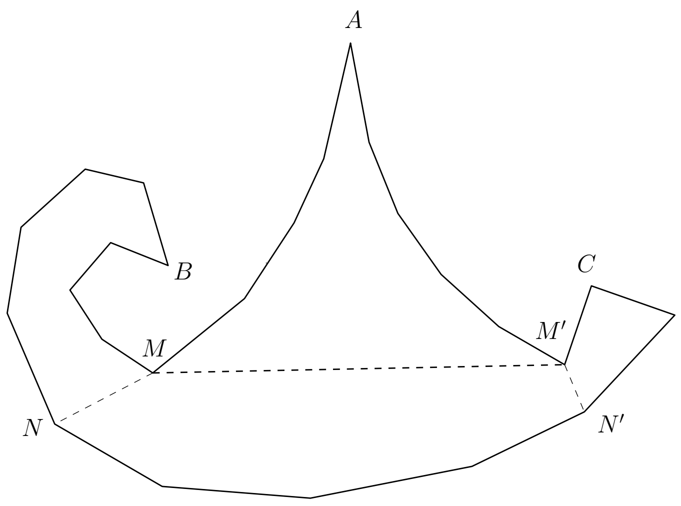

Figure 8: Decomposition of an anchor polygon

Some of the vertices of these spiral sub-polygons may see some of the vertices of the convex one. We extend boundary of these two spiral sub-polygons to $\langle M, ..., B, ..., N, ..., N', M'\rangle$ and $\langle M', ..., C, ..., N', ..., N, M\rangle$, respectively (clearly both of them are spiral polygons yet). These extensions are done to enforce the visibility constraints between the vertices of the convex sub-polygon and the spiral sub-polygons during their reconstruction. The visibility graph of any one of these sub-polygons must satisfy the sub-polygon conditions. Precisely, the induced sub-graph of G on vertices of the tower polygon (resp. spiral polygons) must have necessary conditions of the visibility graph of a tower polygon (resp. spiral polygon) with these boundary vertices, and, the induced sub-graph of G on the convex sub-polygon must be a complete graph. Otherwise, we report that the pair $G(V, E)$ and H does not belong to an anchor polygon with the given joint vertices A, B and C.

4.2 Reconstructing sub-polygons

Now, we are ready to propose the final step of our constructive algorithm for solving both recognizing and reconstruction problems. If we consider the union of the tower and convex sub-polygons as a single polygon, it will be an anchor polygon as well. But, this anchor polygon has this property that its bi-tangent passes through its non-top joint vertices B and C. We call such anchor polygons *simple anchor polygons*. The visibility graph of a simple anchor polygon with joint vertices A, B and C has the following properties.

Observation 9 *The bi-tangent line of an anchor polygon passes through its joint vertices B and C if and only if B and C see each other. In other words, the bi-tangent line passes through B and C if $M = B$ and $M' = C$, i.e. the polygon is a simple anchor polygon. In these cases, the convex chain BC lies completely on the opposite side of the bi-tangent line with respect to A and all of the vertices of this convex chain are visible from each other, so, they form a clique in the visibility graph.*

Observation 10 *As it holds for an anchor polygon, each concave vertex of a simple anchor polygon sees one continuous sub-chain of the convex chain.*

Observation 11 *The joint vertices B and C of a simple anchor polygon see the whole convex chain.*

Observation 12 *For each concave vertex v of a simple anchor polygon, the vertices of the convex chain which are visible to vA^1 are a subset of the vertices visible from v.*

Observation 13 *If both convex vertices p and vB^1 of a simple anchor polygon lie on the right side of the line through A and AB^1, the set of visible concave vertices from p is a subset of this set for pB^1 (see Figure 9.a). Symmetrically, this is true for p and pC^1 if both lie on the left side of the line through A and AC^1.*

Observation 14 *Assume p as a convex vertex on the left side of the line through A and AC^1 in a simple anchor polygon and $q = AB(p)$. Then, none of the vertices of the sub-chain from A to $s = AC(q)A^1$ is visible from p and all vertices of the sub-chain from C to $t = AC(qB^1)$ are visible from p (see Figure 9.b). This means that $AC(p)$ must be one of the vertices of the right concave chain from $AC(q)$ to t.*

Symmetrically, for a convex vertex p lieing on the right side of the line through A and AB^1 where $q = AC(p)$, $AB(p)$ must be one of the vertices of the left concave chain from $AB(q)$ to $AB(qC^1)$.

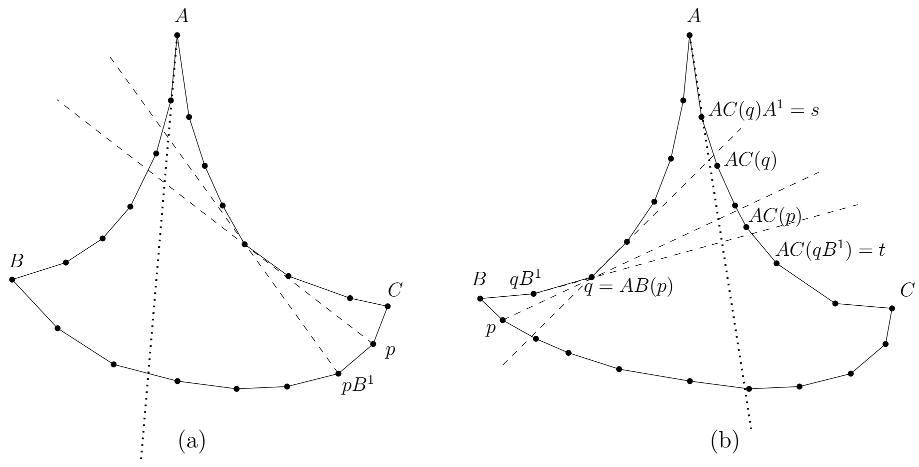

Figure 9: a) Visible vertices from p is a subset of the visible vertices from pB^1 b) Visible and non-visible vertices of concave chain AC from p

Trivially, all above properties must be satisfied on the visibility graph induced to the vertices of the tower and convex sub-polygons in our decomposition algorithm presented in Section 4.1. The details of the verification algorithm for these observations are discussed in Section 5. If these conditions hold, we reconstruct simple anchor polygon which corresponds to the obtained tower and convex sub-polygons.

To reconstruct a simple anchor polygon, we first reconstruct the tower polygon using the method presented in [5] (we use the method as a black-box procedure). Then the vertices of the convex chain are put on a convex curve from B to C supporting their order on the Hamiltonian cycle and the visibility graph constraints. To do this, we divide these vertices into three groups: The first group, called V_A, contains those vertices that see all vertices of both concave chains. From the above observations, these vertices must lie on the convex curve between the lines passing through A and AB^1, and A and AC^1 (See Figure 10.a). The other groups are the sets V_B and V_C as shown in Figure 10a. To locate an arbitrary vertex $v \in V_B$ it must satisfy two conditions: assume that $p = AB(v)$ and $q = AC(v)$. According to the visibility graph constraints, v must lie on the left of the line through p and pA^1 and to the right of the line through p and q. Moreover, v does not see qA^1 and p is a blocking vertex for this non-visibility. Therefore, if qA^1 is visible from p then v must lie to the left of the line through p and qA^1. Otherwise, as v lies to the left of the line through p and pA^1, the vertex p will block the visibility of v and qA^1 and there is no need to add more constraint to restrict position of v on the convex curve. Therefore, the intersection of the convex curve and this region must be non-empty. If this happens, we can put v on an arbitrary point of this part of the convex curve and for all points v that must be located in this region, we put them according to their order in the Hamiltonian cycle. As the last point of our algorithm, we must show that the intersection of the convex curve and the constructed region for v is not empty. The region is restricted to lines d_1 and d_2 or lines d_2 and d_3 (See Figure 10c). It is simple to show that in both cases q is visible from p and in the latter one qA^1 is visible from p and lies above q. This implies that in both cases the region, and consequently, the intersection is not empty. The vertices of V_C are located on the convex curve, symmetrically.

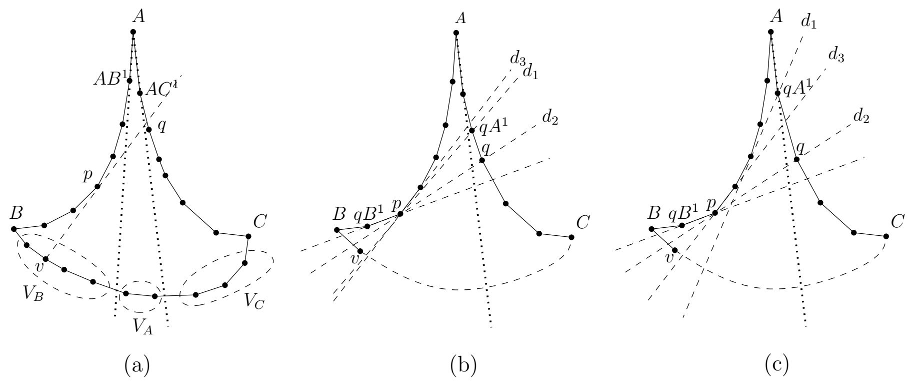

Figure 10: a) V_A, V_B and V_C b) v and qA^1 are non-visible c) non-visibility of v and qA^1 does not force more contraint on the position of v

After reconstructing the simple anchor polygon of the tower and the convex sub-polygons, we must build and attach the spiral sub-polygons to the sides of this simple anchor polygon. Recall that the boundary of the right and left sub-polygons are $\langle M, ..., B, ..., N, ..., N', M'\rangle$ and $\langle M', ..., C, ..., N', ..., N, M\rangle$, respectively. Moreover, remember that in these polygons there is no edge between the vertices of $\langle M, ..., B, ..., N\rangle$ and $\langle M', ..., C, ..., N'\rangle$ except edges that have an end point in $\{N, N', M, M'\}$. This helps to build these parts independently. Note that we can apply this independency by locating the remained vertices of these spiral polygons above the line through M and M'. We describe how to build the left spiral polygon and the right one can be built symmetrically. If we apply the elimination scheme (described in Section 2) starting from the joint vertex B, we find a sequence of removed vertices, which includes all vertices of the left spiral polygon except vertices of the convex sub-polygon, i.e. the elimination scheme on the left spiral sub-polygon stops when the only remaining vertices are the vertices of the convex sub-polygon (which form a clique). Moreover, the remained vertices of this spiral polygon (which make a convex sub-polygon with respect to the Hamiltonian cycle) are the vertices of our convex sub-polygon $\langle M, M', N', N \rangle$, which has been already reconstructed. The spiral polygon reconstruction algorithm described in Section 2 says that we can start from an arbitrary convex polygon for the remained vertices and the sequence of the removed vertices can be put in an arbitrary small neighborhood of the only concave vertex (here it is M). This means that, by considering the convex sub-polygon as the starting convex polygon, we can reconstruct the left spiral polygon arbitrary close to M without intersecting the constructed tower polygon.

5 Complexity Analysis

In this section we analyze the time complexity of our algorithm for recognizing and reconstruction of an anchor polygon from its visibility graph and Hamiltonian cycle.

We use $p^{(i)}$ for the i^{th} vertex after a vertex p in the Hamiltonian cycle and $deg(p)$ for the degree of vertex p in the visibility graph. In order to do the recognition efficiently, for each vertex we need to know its maximal cliques with its previous and successor vertices according to their order in the Hamiltonian cycle, separately. For each vertex p, let $C^+(p)$ (resp. $C^-(p)$) be the maximum size of a subset of vertices consecutively after (resp. before) p, that with p makes a clique in the visibility graph. We calculate these values in $O(|E|)$ time steps, for all vertices using a *Dynamic Programming* [6] algorithm. For this purpose, as shown in Algorithm 1, we begin with calculating $C^+(v)$ for a given vertex v as a pivot. This is done by searching for the first vertex $v^{(i)}$, which is not in a clique with vertices between v and $v^{(i)}$. This search takes $O(\sum_{i=1}^{C^+(v)+1} deg(v^+)) = O(|E|)$ which is overally done by checking all the edges between the vertices from $v^{(1)}$ to $v^{(C^+(v)+1)}$.

Assume that we know the value of $C^+(v)$ for all vertices between v and $v^{(current\_index)}$ in the Hamiltonian cycle, except $v^{(current\_index)}$ itself. Clearly, $C^+(v^{(current\_index)})$ is at least $i = C^+(v^{(current\_index-1)}) - 1$ and we only need to know the maximum size of the set of vertices consecutively after $v^{(C^+(v^{(current\_index-1)})-1)}$, which make a clique with the set of vertices from $v^{(current\_index)}$ to them on the Hamiltonian cycle. We calculate and store the size of this maximal clique, which is $C^+(v)$, iteratively for each vertex $v^{(current\_index)}$ lieing between $v^{(1)}$ and $v^{(|V-1|)}$.

During the execution of Algorithm 1, each edge is checked at most twice, which means that the values of $C^+(u)$ for all vertices $u \in V$ are computed in $O(|E|)$ steps. The values of $C^-(v)$ are computed similarly in $O(|E|)$ steps.

We denote by $D^+(p)$ (resp. $D^-(p)$) the distance between p and the first vertex which is visible to p after (resp. before) its maximal clique. To compute this value for a vertex v, we search for the first vertex after $v^{(C+(v))}$, which is visibile to v. These values can be calculated in $O(deg(v))$ steps for a vertex v and $O(|E|)$ steps for all vertices of G.

At the begining of our recognition algorithm, we check whether the given pair of visibility graph and Hamiltonian cycle belongs to a spiral or tower polygon. These checks needs $O(|E|)$ steps [8, 5]. If it belongs to a tower (resp. spiral) polygon we reconstruct it with the method introduced by Colley *et al.* [5](resp. Everett and Corneil [8]).

Algorithm 1 Calculating C^+

```
1: function $CalcC^+(v, G, H)$(An arbitrary vertex $v$, visibility graph $G$ and its Hamiltonian cycle $H$)
2:     $i \leftarrow 1$
3:     $is\_clique \leftarrow true$
4:     while $i < |V|$ and $is\_clique = true$ do
5:         if $v^{(i)}$ is non-visible to a vertex from $v$ to $v^{(i-1)}$ then
6:             $is\_clique \leftarrow false$
7:         end if
8:         $i \leftarrow i + 1$
9:     end while
10:    $C^+(v) \leftarrow i - 1;$
11:    $current\_index \leftarrow 1$
12:    while $current\_index < |V|$ do
13:        $is\_clique \leftarrow true$
14:        $i \leftarrow C^+(v^{(current\_index)}) - 1$
15:        while $i < |V|$ and $is\_clique = true$ do
16:            if $v^{(current\_index+i)}$ is non-visible to a vertex from $v^{(current\_index)}$ to $v^{(current\_index+i-1)}$ then
17:                $is\_clique \leftarrow false$
18:            end if
19:        end while
20:        $C^+(v^{(current\_index)}) \leftarrow i - 1;$
21:        $current\_index \leftarrow current\_index + 1;$
22:    end while
23:    return $C^+(v)$
24: end function
```

Otherwise, in the first part of our algorithm, we find candidate vertices for the top joint vertex A, formally described in Algorithm 2.

This algorithm iterates on all vertices of H and verify observations 4 and 6 on each of the vertices of G and puts those vertices satisfying these observations in a list, $List$, according to their order in the Hamiltonian cycle (lines 2-11). This part of the algorithm needs $O(|E|)$ steps because we meet each edge of the visibility graph at most twice. For each vertex A in this list, we call Algorithm 3 to find its corresponding other joint vertices B_A and C_A. Observation 1 implies that B_A lies between A and the next vertex after A in $List$. Therefore, in all calls of Algorithm 3, each vertex is processed at most twice. Therefore, all these calls of Algorithm 3 are done in time $O(|E|)$. In order to verify Observation 3 for a top joint vertex A, we start from $v = B_A$ toward C_A. If $v^{(D^+(v))}$ or $v^{(-D^-(v))}$ belongs to B_AC_A for a vertex v, then Observation 3 is not verified and A is not a candidate for top joint vertex. If we find a vertex v like that, we stop checking other vertices and reject A (lines 19-33).

Algorithm 2 Finding top joint vertex candidates

```
function FindA(Visibility graph $G$ and its Hamiltonian cycle $H$)
    $PreList \leftarrow$ an empty list
    $v \leftarrow$ an arbitrary vertex in $V(G)$
    for $i$ from 0 to $|V|$ do
        if $(v^{(i+1)}, v^{(i-1)}) \in E(G)$ then
            if $(v^{(i+1)}, v^{(i+2)}) \notin E(G)$ and $(v^{(i-1)}, v^{(i-2)}) \notin E(G)$ then
                Add $v^{(i)}$ to the end of $PreList$
            end if
        end if
    end for
    $List \leftarrow PreList$ as an array
    $n \leftarrow$ Number of elements in $List$
    if $n = 0$ then return $\{\}$
    end if
    $Cadidates \leftarrow$ an empty list
    $c \leftarrow 1$
    $A \leftarrow List[1]$
    $LastChecked \leftarrow A^{(2)}$
    repeat
        $A \leftarrow List[c]$
        $c \leftarrow c + 1$
        $i \leftarrow 0$
        $(B_A, C_A) \leftarrow FindBC(A)$
        if $LastChecked$ lies between $A$ and $B_A$ then
            $LastChecked \leftarrow B_A$
        end if
        $valid \leftarrow true$
        while $B_A^{(i)} \neq C_A$ do
            if $D^+(LastChecked)$ lies before $C_A$ or $D^-(LastChecked)$ lies after $C_A$ then
                $valid \leftarrow false$
            end if
            $LastChecked \leftarrow LastChecked^{(1)}$
            $i \leftarrow i + 1$
        end while
        if $valid = true$ then
            Add $(A, B_A, C_A)$ into $Candidates$
        end if
    until $c \leq n$
    return $Candidates$
end function
```

In line 20 we assign $List[c]$ to the variable A. At this point, assume that A' is the next vertex in $List$ after A, i.e. $List[c+1]$. Therefore, it will be checked for whether it can be a candidate for top vertex in the next iteration of the loop after A. Line 23 computes the joint vertices B_A and C_A for the given candidate for top vertex A. Assume that in the iteration of the loop, $B_{A'}$ and $C_{A'}$ are going to be computed as the joint vertices of A'. Observe that if we reject A because of a vertex $LastChecked$ in line 29, we do not need to check vertices between B_A and $LastChecked$ to see, whether they satisfy Observation 3, and we only need to check vertices between $LastChecked$ and $B_{A'}$. Therefore, we check each vertex of $V(G)$ in the process of verifying Observation 3 at most once and this part of the algorithm is done in $O(|E|)$ steps and we find candidate vertices for top joint vertex and their corresponding joint vertices in $O(|E|)$ steps. It remains to verify observations 7 to 14 to complete the recognition.

Algorithm 3 Finding joint vertices B_A and C_A of a top joint vertex A

```
1: function FindBC(An arbitrary vertex $A$, visibility graph $G$ and its Hamiltonian cycle $H$)
2:     $i \leftarrow 1$
3:     while $i < |V|$ and $(A^{(i-1)}, A^{(i+1)}) \notin E(G)$ do
4:         $i \leftarrow i + 1$
5:     end while
6:     $j \leftarrow -1$
7:     while $(-j) < |V|$ and $(A^{(j-1)}, A^{(j+1)}) \notin E(G)$ do
8:         $j \leftarrow j - 1$
9:     end while
10:    return $(A^{(i)}, A^{(j)})$
11: end function
```

The algorithm in Section 4.1 takes at most $O(|E|)$ steps to verify observations 7 and 8 to find the bi-tangent line and decompose polygon into a simple anchor polygon and two spiral polygons. The simple anchor polygon must satisfy observations 10 to 14. To verify this, we first compute $AC(v)$, $CA(v)$, $AB(v)$, $BA(v)$, $BC(v)$ and $CB(v)$ for a vertex v in $O(deg(v))$ steps. We can compute these values for all vertices in $O(|E|)$ steps from which, observations 10 to 14 can be verified in $O(|E|)$ steps. Then, the tower polygon of the simple anchor polygon is reconstructed in $O(|E|)$ steps [5] and its convex chain is reconstructed as described in Section 4.2 and depicted in Fig 10 in $O(|E|)$ steps as well. Finally, recognizing and reconstruction of each spiral polygon requires $O(|E|)$ steps [8] steps. Therefore, our algorithm will recognize and reconstruct an anchor polygon in $O(|E|)$ steps. All the steps in our method are composed of basic arithmetic and constant time operations. Therefore, our algorithm will recognize and reconstruct an anchor polygon in $O(|E|)$ time which is linear in terms of the input size.

6 Conclusions

In this paper, we proposed a method for recognizing and reconstructing an anchor polygon from its visibility graph and Hamiltonian cycle. This was done in linear time in terms of the input size. As the problem is still open for general simple polygons, this result along with previous results for tower and spiral polygons are steps toward attacking the main problem. Other that trying to solve this problem for general polygons, other future research directions are extending these results to other special polygons like 2-spiral or polygons with constant number of chains, and trying to relate the complexity of the problem to the number of chains on the polygon boundary.

References

[1] J. Abello and K. Kumar. Visibility graphs and oriented matroids (extended abstract). In R. Tamassia and I. G. Tollis, editors, *Graph Drawing*, pages 147–158, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. doi:10.1007/3-540-58950-3_366.

[2] J. Abello and K. Kumar. Visibility graphs of 2-spiral polygons (extended abstract). In R. Baeza-Yates, E. Goles, and P. V. Poblete, editors, *LATIN '95: Theoretical Informatics*, pages 1–15, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. doi:10.1007/3-540-59175-3_77.

[3] J. Abello, H. Lin, and S. Pisupati. On visibility graphs of simple polygons. *Congressus Numerantium*, pages 119–119, 1992.

[4] C. Belta, V. Isler, and G. Pappas. Discrete abstractions for robot motion planning and control in polygonal environments. *IEEE Transactions on Robotics*, 21(5):864–874, 2005. doi:10.1109/TRO.2005.851359.

[5] P. Colley, A. Lubiw, and J. Spinrad. Visibility graphs of towers. *Computational Geometry*, 7(3):161–172, 1997. doi:10.1016/0925-7721(95)00033-X.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3rd edition.

[7] H. Everett. Visibility graph recognition, 1990. Phd. Dissertation, University of Toronto.

[8] H. Everett and D. Corneil. Recognizing visibility graphs of spiral polygons. *Journal of Algorithms*, 11(1):1–26, 1990. doi:10.1016/0196-6774(90)90026-B.

[9] S. K. Ghosh. On recognizing and characterizing visibility graphs of simple polygons. In R. Karlsson and A. Lingas, editors, *SWAT 88*, pages 96–104, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg. doi:10.1007/3-540-19487-8_10.

[10] S. K. Ghosh. On recognizing and characterizing visibility graphs of simple polygons. *Discret. Comput. Geom.*, 17(2):143–162, 1997. doi:10.1007/BF02770871.

[11] S. K. Ghosh. *Visibility algorithms in the plane*. Cambridge University Press, 2007.

[12] J. Richter-Gebert. Mnëv's universality theorem revisited. *Séminaire Lotaringien de Combinatorie*, 1995.

[13] I. Streinu. Non-stretchable pseudo-visibility graphs. *Computational Geometry*, 31(3):195–206, 2005. doi:10.1016/j.comgeo.2004.12.003.

[14] S. Teller and P. Hanrahan. Global visibility algorithms for illumination computations. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, page 239–246, New York, NY, USA, 1993. Association for Computing Machinery. doi:10.1145/166117.166148.