Reconstructing a Simple Polygon from Its Angles

Yann Disser, Matúš Mihalák, and Peter Widmayer

Institute of Theoretical Computer Science, ETH Zürich
{ydisser,mmihalak,widmayer}@inf.ethz.ch

Abstract. We study the problem of reconstructing a simple polygon from angles measured at the vertices of the polygon. We assume that at each vertex, a sensing device returns the sequence of angles between each pair of vertices that are visible. We prove that the sequence of angle measurements at all vertices of a simple polygon in cyclic order uniquely determines the polygon up to similarity. Furthermore, we propose an algorithm that reconstructs the polygon from this information in polynomial time.

1 Introduction

The reconstruction of geometric objects from measurement data has attracted considerable attention over the last decade [7,11,13]. In particular, many variants of the problem of reconstructing a polygon with certain properties have been studied. For different sets of data this polygon reconstruction problem has been shown to be NP-hard [4,8,10]. Recently, data from rather novel sensing devices like range-finding scanners has been considered, and most of the reconstruction problems that such devices naturally induce have been shown to be NP-hard as well, while a few others are polynomial time solvable [1]. We study the reconstruction problem induced by sensors that measure a sequence of angles. Specifically, we assume that at each vertex v of a simple polygon, the sequence of vertices *visible* from v is perceived in counterclockwise (ccw) order as seen around v, starting at the ccw neighbor vertex of v on the polygon boundary. As usual, we call two polygon vertices (mutually) *visible*, if the straight line segment connecting them lies entirely in the polygon. In addition to seeing visible vertices the angle sensor measures angles between adjacent rays from v to the vertices it sees, and it returns the ccw sequence of these measured angles (cf. Figure 1). Note that such an angle measurement indirectly also yields the angles between any pair of rays to visible vertices (not only adjacent pairs). Our polygon reconstruction problem takes as input a ccw sequence of angle measurements, one measurement at each vertex of a simple polygon, and asks for a simple polygon that fits the measured angles; we call this problem the *polygon reconstruction problem from angles* (cf. Figure 2).

Our contribution. We propose an algorithm that solves the polygon reconstruction problem from angles in polynomial time, and we show that the solution is unique (up to similarity). More precisely, we focus on the visibility graph, i.e.,

H. Kaplan (Ed.): SWAT 2010, LNCS 6139, pp. 13–24, 2010.

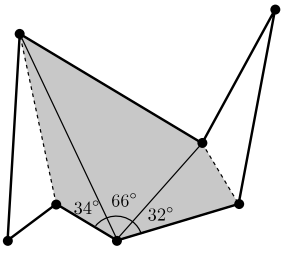

Fig. 1. Illustration of an angle measurement: the sensor returns the vector $(32°, 66°, 34°)$

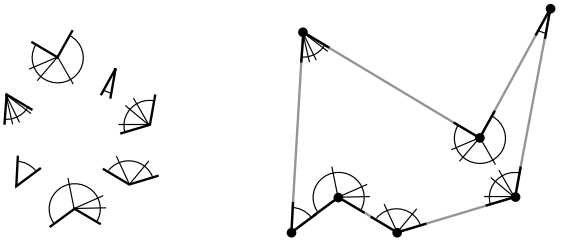

Fig. 2. Given a sequence of angle measurements in ccw order along the boundary (left); the goal is to find a polygon that fits these angles (right)

the graph with a node for every vertex of the polygon and an edge between two nodes if the corresponding vertices see each other. It is sufficient to reconstruct the visibility graph of a polygon, as, together with the angle data, it is then easy to infer the shape of the polygon up to similarity.[1] We show that only the visibility graph of the original polygon $\mathcal{P}$ is compatible with the information contained in the angle data measured in $\mathcal{P}$. Our algorithm finds this unique visibility graph in polynomial time and thus reconstructs the original polygon up to similarity in polynomial time. Note that if only the set of angle measurements is given, i.e. the order of the vertices along the boundary is not known, it is impossible to uniquely reconstruct the visibility graph of a polygon in general.[2] While we assume that the measured angles come from a simple polygon, our algorithm as a side effect is also capable of detecting false inputs, i.e., measurements that do not fit any simple polygon.

[1] The shape of the polygon can be obtained in linear time from the visibility graph and angle data. We can achieve this by first computing a triangulation and then fixing the length of one edge. All other lengths in the triangulation can then be computed in linear time.

[2] To see this, consider a square and "attach" to every corner of it a triangle. Make the shapes of the triangles all different and such that the vertices of a triangle that are not at the corner of the square only see the corner vertices of the square they are attached to (plus the vertices of the triangle of course). Now any permutation of the triangles results in the same set of angle measurements but in different polygons.

The key difficulty of the reconstruction of the visibility graph lies in the fact that vertices in our setting have no recognizable labels, i.e., an angle measurement at a vertex returns angles between distant vertices but does not identify these distant vertices globally. Instead, our algorithm needs to identify these vertices in a consistent way across the whole input. In this sense, our problem has a similar flavor as the turnpike reconstruction problem (also known as one dimensional partial digest problem), whose complexity is still open [12].

Related Work. For our purposes, the combinatorial nature of a polygon is encoded in its visibility graph. Solving the visibility graph reconstruction problem for certain data may be a step towards understanding visibility graphs in general. Their characterization has been an open problem for many years that has attracted considerable attention [5,6].

A question closely related to the offline reconstruction of the visibility graph of a polygon appears in the area of robotic exploration, namely what sensory and motion capabilities enable simple robots inside a polygon to reconstruct the visibility graph [2,14]. The idea to reconstruct it from angle data was first mentioned in this context [2], but was also discussed earlier [9]. In all these models a simple robot is assumed to sense visible vertices in ccw order (but does not sense the global identity of visible vertices). In [2], the problem of reconstructing the visibility graph of a polygon was solved for simple robots that can measure angles and additionally are equipped with a compass. In the case of robots that can only distinguish between angles smaller and larger than π, it was shown in the same study that adding the capability of retracing their movements empowers the robots to reconstruct the visibility graph (even if they do not know n, the number of vertices of the unknown polygon). In both cases a polynomial-time algorithm was given. Recently, it was shown that the ability to retrace their movements alone already enables simple robots to reconstruct the visibility graph (when at least an upper bound on the number of vertices of the polygon is given), even though only an exponential algorithm was given [3]. Our result implies that measuring angles alone is also already sufficient. On the other hand, it is known that the inner angles (the angles along the boundary) of the polygon do not contain sufficient information to uniquely reconstruct the visibility graph, even when combined with certain combinatorial information [2].

The general problem of reconstructing polygons from measurement data has mainly been studied in two variants. The first variant asks to find some polygon $\mathcal{P}^\star$ that is consistent with the data measured in the original polygon $\mathcal{P}$. For example, it was studied how a polygon $\mathcal{P}^\star$ compatible with the information obtained from "stabbing" $\mathcal{P}$ or compatible with the set of intersection points of $\mathcal{P}$ with some lines can be constructed [7,11]. The problem we consider falls in the second variant in which generally the problem is to reconstruct $\mathcal{P}$ itself uniquely from data measured in $\mathcal{P}$, i.e., we have to show that only $\mathcal{P}$ is compatible with the data. A previous study in this area shows that the inner angles of $\mathcal{P}$ together with the cross-ratios of its triangulation uniquely determine $\mathcal{P}$ [13].

Outline. We introduce the visibility graph reconstruction problem in detail in Section 2. In Section 3 we show that there is a unique solution to the problem, and give an algorithm that finds the unique solution in polynomial time.

2 The Visibility Graph Reconstruction Problem

Let $\mathcal{P}$ be a simple polygon with visibility graph $G_{\text{vis}} = (V, E_{\text{vis}})$, where V denotes the set of vertices of $\mathcal{P}$ and $n = |V|$. We fix a vertex $v_0 \in V$ and denote the other vertices of $\mathcal{P}$ by $v_1, v_2, \ldots, v_{n-1}$ in ccw order along the boundary starting at v_0's ccw neighbor. For ease of presentation only, we assume polygons to be in general position, i.e. no three vertices are allowed to lie on a line. All definitions and results can be adapted to be valid even without this assumption (note that our definition of *visible* vertices implies that the line segment connecting two mutually visible vertices can have more than two points on the boundary of the polygon in this case). The degree of a vertex $v_i \in V$ in G_{vis} is denoted by $d(v_i)$ and the sequence $\text{vis}(v_i) = \left(v_i, u_1, u_2, \ldots, u_{d(v_i)}\right)$ is defined to enumerate the vertices visible to v_i ordered in ccw order along the boundary starting with v_i itself. We write $\text{vis}_0(v_i)$ to denote v_i itself and $\text{vis}_k(v_i)$, $1 \leq k \leq d(v_i)$ to denote u_k. For two distinct vertices $v_i, v_j \in V$, $\text{chain}(v_i, v_j)$ denotes the sequence $(v_i, v_{i+1}, \ldots, v_j)$ of the vertices between v_i and v_j along the boundary in ccw order. Similarly, $\text{chain}_v(v_i, v_j)$ denotes the subsequence of $\text{chain}(v_i, v_j)$ that contains only the vertices that are visible to v. Note that here and in the following all indices are understood modulo n.

We define the *visibility segments* of v to be the segments $\overline{vu_1}, \overline{vu_2} \ldots, \overline{vu_{d(v)}}$ in this order. Similarly, we define the *visibility angles* of v to be the ordered sequence of angles between successive visibility segments, such that the i-th visibility angle is the angle between $\overline{vu_i}$ and $\overline{vu_{i+1}}$, for all $1 \leq i \leq d(v) - 1$.

Let $v, v_i, v_j \in V$. We write $\measuredangle_v(v_i, v_j)$ to denote the angle between the lines $\overline{vv_i}$ and $\overline{vv_j}$ (in that order) even if v, v_i, v_j do not mutually see each other. Let $1 \leq l < r \leq d(v)$. We write $\angle_v(l, r)$ to denote $\measuredangle_v(\text{vis}_l(v), \text{vis}_r(v))$. We will need the notion of the approximation $\measuredangle_v^{\uparrow}(v_i, v_j)$ of the angle $\measuredangle_v(v_i, v_j)$, which is defined as follows (cf. Figure 3): Let $v_{i'}$ be the last vertex in $\text{chain}_v(v_{i+1}, v_i)$ and $v_{j'}$ be the first vertex in $\text{chain}_v(v_j, v_{j-1})$. We then define $\measuredangle_v^{\uparrow}(v_i, v_j) = \measuredangle_v(v_{i'}, v_{j'})$. Observe that if $\{v, v_i\}, \{v, v_j\} \in E_{\text{vis}}$, we have $\measuredangle_v^{\uparrow}(v_i, v_j) = \measuredangle_v(v_i, v_j)$. Also note that knowing the visibility angles of a vertex v is equivalent to knowing $\angle_v(l_v, r_v)$ for all $1 \leq l_v < r_v \leq d(v)$.

In terms of the above definitions, the goal of the visibility graph reconstruction problem is to find E_{vis} when we are given n, V, $d(v)$ for all $v \in V$, and $\angle_v(l_v, u_v)$ for all $v \in V$ and all $1 \leq l_v < u_v \leq d(v)$, as well as the (ccw) order in which the vertices appear along the boundary.

3 Triangle Witness Algorithm

The key question when trying to reconstruct the visibility graph of a polygon is how to identify a vertex u visible to some known vertex v. Knowing all angles at

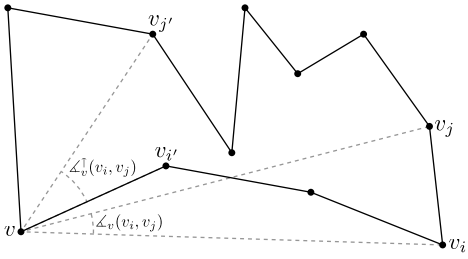

Fig. 3. Illustration of the approximation $\measuredangle_v^{\uparrow}(v_i, v_j) = \measuredangle_v(v_{i'}, v_{j'})$ of the angle $\measuredangle_v(v_i, v_j)$

every vertex may seem to be far too much information and the reconstruction problem may thus seem easily solvable by some greedy algorithm. Before we actually present the triangle witness algorithm that solves the reconstruction problem, we show that some natural greedy algorithms do not work in general.

Greedy Approach. It is a natural idea to first orient all angles w.r.t. a single, global orientation (e.g. the line $\overline{v_{n-1}v_0}$) by summing angles around the polygon boundary. Then, if a vertex v sees some other vertex u under a certain global angle α, u must see v under the inverse angle $\alpha + \pi$, as the line $\overline{uv}$ has a single orientation. A simple greedy approach to identify the vertex u in the view from v could be to walk from v along the boundary and find the first vertex that sees some other vertex under the global angle $\alpha + \pi$. The example in Fig. 4 however shows that this approach does not work in general.

A similar but somewhat stronger approach is to allow global angles to go beyond $[0, 2\pi)$ while summing up around the polygon boundary, cf. Figure 4 (there, for instance, vertex v_1 sees the second visible vertex in ccw order under the angle $\alpha - \pi$ which is less than 0). This would prevent the pairing of vertex v_0 with vertex v_1 in that example. Nevertheless, there are still examples where this strategy fails and in fact it is not possible to greedily match angles:[3] inspect Figure 5 for an example of two polygons for which no matter how a greedy algorithm chooses to pair vertices, it has to fail for one of the two.

Triangle Witness Algorithm. We now give an algorithm for the reconstruction of the visibility graph from the visibility angles of all vertices. Note that from now on we map all angles to the range $[0, 2\pi)$. Our algorithm considers all vertices at once and gradually identifies edges connecting vertices that lie further and further apart along the boundary. Intuitively, once we know all vertices in $\{v_{i+1}, v_{i+2}, \ldots, v_{k-1}\}$ that are visible to v_i, there is only one candidate vertex which might be v_k, namely the next unidentified vertex in $\text{vis}(v_i)$. Our algorithm thus only needs to decide whether v_i sees v_k. The key ingredient here is the use of a *triangle witness* vertex that indicates whether two other vertices see each other. Because any polygon can be triangulated, we know that for every two vertices $\{v_i, v_j\} \in E_{\text{vis}}$ with $v_j \neq v_{i+1}$, there is a "witness" vertex $v_l \in \text{chain}(v_{i+1}, v_{j-1})$

[3] We do not aim, however, to give complete proof or to fully characterize all failing greedy algorithms based on the idea of angle matching.

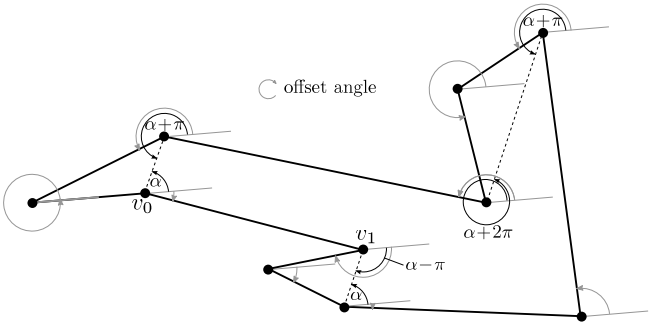

Fig. 4. Illustration of the idea behind the greedy pairing algorithm for a single angle α and starting vertex v_0. If we map angles to the range $[0, 2\pi)$, we allow v_0 and v_1 to be paired which is obviously impossible.

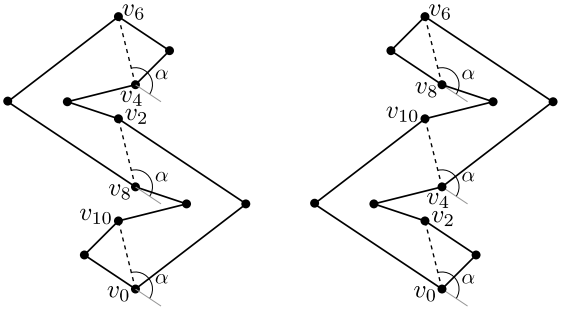

Fig. 5. An example in which only one visibility graph can correctly be reconstructed by any greedy pairing algorithm

that they both see, such that v_i, v_l, v_j form a triangle (of angle sum π). We now extend this notion to the case where $\{v_i, v_j\} \notin E_{\text{vis}}$.

Definition 1. *Let $v_i, v_j \in V$ be two different vertices and $v_j \neq v_{i+1}$. Let further $v_l \in \text{chain}(v_{i+1}, v_{j-1})$ with $\{v_i, v_l\}, \{v_j, v_l\} \in E_{\text{vis}}$. Then we say v_l is a* triangle witness *of (v_i, v_j), if it fulfills the* generalized angle-sum condition

$$\measuredangle^{\uparrow}_{v_i}(v_l, v_j) + \measuredangle^{\uparrow}_{v_j}(v_i, v_l) + \measuredangle_{v_l}(v_j, v_i) = \pi.$$

In the following we motivate the definition of a triangle witness (cf. Figure 6). As before, we know that if two vertices $v_i, v_j \in V, v_j \neq v_{i+1}$ see each other, there must be a vertex $v_l \in \text{chain}(v_{i+1}, v_{j-1})$ which sees both of them. For any choice of v_l, the condition $\measuredangle_{v_i}(v_l, v_j) + \measuredangle_{v_j}(v_i, v_l) + \measuredangle_{v_l}(v_j, v_i) = \pi$ is trivially fulfilled. In the case that v_i does not see v_j, the only difference from v_i's perspective is that for any choice of v_l, $\measuredangle_{v_i}(v_l, v_j)$ does not appear in its visibility angles. We want to modify the condition to capture this difference. The idea is to replace v_j in $\measuredangle_{v_i}(v_l, v_j)$ by an expression that happens to be v_j, if and only if v_i sees v_j. We choose "the first vertex in $\text{chain}_{v_i}(v_j, v_{j-1})$", which is v_j, exactly if v_i sees v_j. If,

similarly, we also replace v_i in $\measuredangle_{v_j}(v_i, v_l)$ by "the last vertex in $\text{chain}_{v_j}(v_{i+1}, v_i)$", we obtain the generalized angle-sum condition of Definition 1. We will later see (stated as Lemma 4) that there is a triangle witness for a pair (v_i, v_j), if and only if $\{v_i, v_j\} \in E_{\text{vis}}$.

We can now describe the triangle witness algorithm. It iterates through increasing number of steps k along the boundary, focusing at step k on all edges of the form $\{v_i, v_{i+k}\}$. Throughout it maintains two maps F, B that store for every vertex all the edges identified so far that go at most k steps forward or backward along the boundary, respectively. We write $F[v_i][v_j] = s$, if $\{v_i, v_j\}$ is the s-th edge incident to v_i in ccw order, and $B[v_i][v_j] = s$, if $\{v_i, v_j\}$ is the s-th edge incident to v_i in ccw order. Note that $B[v_i]$ is filled in cw order during the algorithm, i.e. its first entry will be $B[v_i][v_{i-1}] = d(v_i)$. Whenever convenient, we use $F[v_i]$ and $B[v_i]$ like a set, e.g. we write $v_l \in F[v_i]$ to denote that there is an entry v_l in $F[v_i]$ and write $|F[v_i]|$ to denote the number of entries in $F[v_i]$. Observe also that $|F[v_i]| + 1$ is the index of the first vertex (in ccw order) in $\text{vis}(v_i)$ that is not yet identified. It is clear that once we completed the maps for k up to $\lceil \frac{n}{2} \rceil$, we essentially have computed E_{vis}.

The initialization of the maps for $k = 1$ is simple as every vertex sees its neighbors on the boundary. In later iterations for every vertex v_i there is always exactly one candidate vertex for v_{i+k}, namely the $(|F[v_i]| + 1)$-th vertex visible to v_i. We decide whether v_i and v_{i+k} see each other by going over all vertices between v_i and v_{i+k} in ccw order along the boundary and checking whether there is a triangle witness $v_l \in \text{chain}(v_{i+1}, v_{i+k-1})$. If and only if this is the case, we update E_{vis}, F, B with the edge $\{v_i, v_{i+k}\}$. For a listing of the triangle witness algorithm see Algorithm 1.

In the following we prove the correctness of the triangle witness algorithm. For this we mainly have to show that having a triangle witness is necessary and sufficient for a pair of vertices to see each other. To show this, we will need the notion of *blockers* and *shortest paths* in polygons.

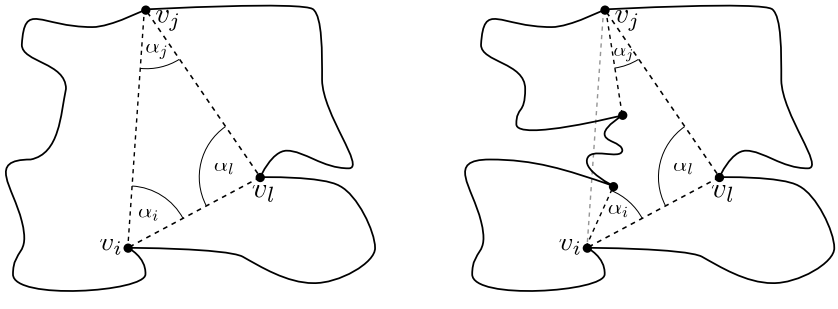

Fig. 6. Illustration of the generalized angle sum condition of Definition 1. On the left $\{v_i, v_j\} \in E_{\text{vis}}$ and the angles α_i, α_j and α_l of the condition sum up to π. On the right, $\{v_i, v_j\} \notin E_{\text{vis}}$ and the sum of the angles is strictly less than π.

Algorithm 1. Triangle witness algorithm

input: n, $d(\cdot)$, $\angle.(\cdot,\cdot)$
output: E_{vis}

1. $F \leftarrow$ [array of n empty maps], $B \leftarrow$ [array of n empty maps], $E_{\text{vis}} \leftarrow \emptyset$
2. **for** $i \leftarrow 0, \ldots, n-1$
3. $\quad E_{\text{vis}} \leftarrow E_{\text{vis}} \cup \{v_i, v_{i+1}\}$
4. $\quad F[v_i][v_{i+1}] \leftarrow 1$
5. $\quad B[v_{i+1}][v_i] \leftarrow d(v_i)$
6. **for** $k \leftarrow 2, \ldots, \lceil \frac{n}{2} \rceil$
7. $\quad$ **for** $i \leftarrow 0, \ldots, n-1$
8. $\quad\quad j \leftarrow i + k$
9. $\quad\quad$ **for** $l \leftarrow i+1, \ldots j-1$
10. $\quad\quad\quad$ **if** $v_l \in F[v_i] \wedge v_l \in B[v_j]$
11. $\quad\quad\quad\quad \alpha_i \leftarrow \angle_{v_i}(F[v_i][v_l], |F[v_i]| + 1)$ $\quad (= \measuredangle^{\uparrow}_{v_i}(v_l, v_j)$, cf. proof of Th. 1)
12. $\quad\quad\quad\quad \alpha_j \leftarrow \angle_{v_j}(d(v_j) - |B[v_j]|, B[v_j][v_l])$ $\quad (= \measuredangle^{\uparrow}_{v_j}(v_i, v_l)$, cf. proof of Th. 1)
13. $\quad\quad\quad\quad \alpha_l \leftarrow \angle_{v_l}(F[v_l][v_j], B[v_l][v_i])$ $\quad (= \measuredangle_{v_l}(v_j, v_i)$, cf. proof of Th. 1)
14. $\quad\quad\quad\quad$ **if** $\alpha_i + \alpha_j + \alpha_l = \pi$
15. $\quad\quad\quad\quad\quad E_{\text{vis}} \leftarrow E_{\text{vis}} \cup \{v_i, v_j\}$
16. $\quad\quad\quad\quad\quad F[v_i][v_j] = |F[v_i]| + 1$
17. $\quad\quad\quad\quad\quad B[v_j][v_i] = d(j) - |B[v_j]|$
18. $\quad\quad\quad\quad\quad$ abort innermost loop

Definition 2. *Let $v_i, v_j \in V$. We say $v_b \in \text{chain}(v_{i+1}, v_{j-1})$ is a* blocker *of (v_i, v_j), if for all $u \in \text{chain}(v_i, v_{b-1}), v \in \text{chain}(v_{b+1}, v_j)$ we have $\{u, v\} \notin E_{\text{vis}}$ (cf. Figure 7 (left)).*

Note that if v_b is a blocker of (v_i, v_j), v_b also is a blocker of (u, v) for all $u \in \text{chain}(v_i, v_{b-1})$, $v \in \text{chain}(v_{b+1}, v_j)$.

A *path* between two vertices $a, b \in V$ of a polygon $\mathcal{P}$ is defined to be a curve that lies entirely in $\mathcal{P}$ and has a and b as its endpoints. A *shortest path* between a and b is a path of minimum Euclidean length among all the paths between the two vertices.

Lemma 1 (Lemmas 3.2.3 and 3.2.5. in [5]). *Let $v_i, v_j \in V$. The shortest path between v_i and v_j is unique and is a chain of straight line segments that connect at vertices.*

We can therefore write $(a, u_0, u_1, \ldots, b)$ to denote a shortest path, where we refer to the u_i's as the path's *interior vertices*. The following statements motivate the term 'blocker'.

Lemma 2. *Let $v_i, v_j \in V$ with $\{v_i, v_j\} \notin E_{\text{vis}}$. Every interior vertex of the shortest path from v_i to v_j is a blocker of either (v_i, v_j) or (v_j, v_i).*

Proof. Consult Figure 7 (right) along with the proof. For the sake of contradiction assume that $v_b \in V$ is an interior vertex of the shortest path p_{ij} from v_i to v_j that is not a blocker of either (v_i, v_j) or (v_j, v_i). W.l.o.g. assume

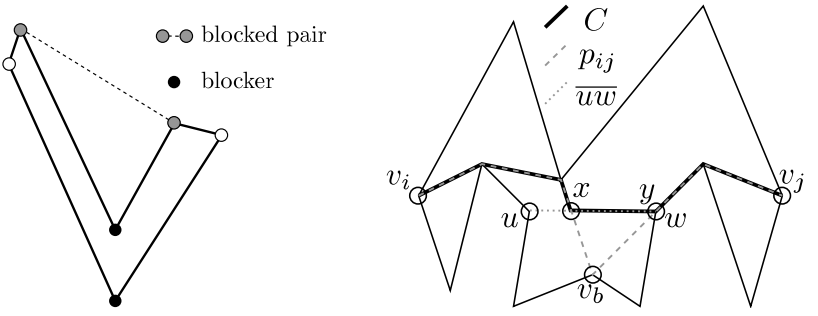

Fig. 7. Left: a pair of vertices can have blockers on both sides. Right: illustration of the objects in the proof of Lemma 2.

$v_b \in \text{chain}(v_{i+1}, v_{j-1})$. As v_b is not a blocker of (v_i, v_j), there are two vertices $u \in \text{chain}(v_{i+1}, v_{b-1}), w \in \text{chain}(v_{b+1}, v_{j-1})$ with $\{u, w\} \in E_{\text{vis}}$. Thus, the segment $\overline{uw}$ is entirely in the polygon and separates it in two parts, one part containing v_b and the other containing both v_i and v_j. As p_{ij} visits v_b, it must cross $\overline{uw}$ at least twice. Let x, y be the first and last intersection points of $\overline{uw}$ with p_{ij}. Consider the curve C that follows p_{ij} until x, then follows $\overline{uw}$ until y and finally follows p_{ij} until v_j. Because of the triangle inequality, C is strictly shorter than p_{ij} which is a contradiction with the assumption that p_{ij} is a shortest path. □

Corollary 1. *Let $v_i, v_j \in V$. If $\{v_i, v_j\} \notin E_{\text{vis}}$, there is either a blocker of (v_i, v_j) or of (v_j, v_i).*

We now relate the definition of a blocker to the geometry of the polygon.

Lemma 3. *Let $v_i, v_j \in V$ with $i = j + 2$, $\{v_i, v_j\} \notin E_{\text{vis}}$. If $w := v_{j+1} = v_{i-1}$ is convex (inner angle $< \pi$), then $v_{i'} = \arg\min_{v_b \in \text{chain}_{v_i}(v_{i+1}, v_{j-1})} \measuredangle_{v_i}(v_b, w)$ and $v_{j'} = \arg\min_{v_b \in \text{chain}_{v_j}(v_{i+1}, v_{j-1})} \measuredangle_{v_j}(w, v_b)$ are blockers of (v_i, v_j) that lie left of the oriented line $\overrightarrow{v_i v_j}$.*

Proof. As w is convex, the shortest path p_{ij} from v_i to v_j only contains vertices of $\text{chain}(v_i, v_j)$. As p_{ij} only makes right turns (i.e. any three consecutive vertices on p_{ij} form a ccw triangle), all interior vertices of p_{ij} lie left of the oriented line $\overrightarrow{v_i v_j}$. Furthermore $v_{i'}$ and $v_{j'}$ are the first and the last interior vertices of p_{ij} respectively. By Lemma 2 we thus know that both $v_{i'}$ and $v_{j'}$ are blockers of (v_i, v_j). From before we also know that they both lie left of the oriented line $\overrightarrow{v_i v_j}$. □

We now get to the central lemma that essentially states that the existence of a triangle witness is necessary and sufficient for a pair of vertices to see each other.

Lemma 4. *Let $v_i, v_j \in V$ with $|\text{chain}(v_i, v_j)| > 2$. There is a triangle witness v_l for (v_i, v_j), if and only if $\{v_i, v_j\} \in E_{\text{vis}}$.*

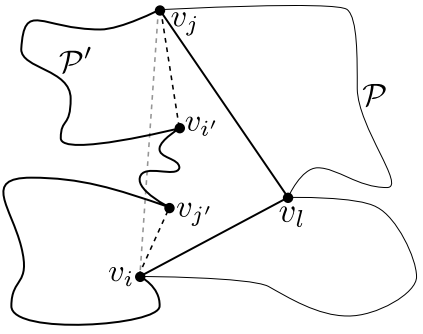

Fig. 8. Sketch of the definitions in the proof of Lemma 4

Proof. If $\{v_i, v_j\} \in E_{\text{vis}}$, because any polygon can be triangulated, there must be a vertex $v_l \in \text{chain}(v_{i+1}, v_{j-1})$ for which both edges $\{v_i, v_l\}$ and $\{v_l, v_j\}$ are in E_{vis}. For this vertex we have $\measuredangle^{\uparrow}_{v_i}(v_l, v_j) + \measuredangle^{\uparrow}_{v_j}(v_i, v_l) + \measuredangle_{v_l}(v_j, v_i) = \measuredangle_{v_i}(v_l, v_j) + \measuredangle_{v_j}(v_i, v_l) + \measuredangle_{v_l}(v_j, v_i) = \pi$ as all three relevant edges are in E_{vis} and the sum over the angles of any triangle is π.

For the converse implication assume there is a triangle witness v_l of (v_i, v_j). For the sake of contradiction, assume $\{v_i, v_j\} \notin E_{\text{vis}}$.

Consider the polygon $\mathcal{P}'$ induced by the vertices $v_i, v_l, v_j, \text{chain}(v_{j+1}, v_{i-1})$, cf. Figure 8. As $\{v_i, v_l\}, \{v_l, v_j\} \in E_{\text{vis}}$, $\mathcal{P}'$ is simple and well defined. In $\mathcal{P}'$, v_l is a convex vertex, as it fulfills the generalized angle-sum condition of Definition 1 and thus $\measuredangle_{v_l}(v_j, v_i) < \pi$, because all angles are positive. We can therefore apply Lemma 3 (on v_j, v_i) w.r.t. $\mathcal{P}'$ and conclude that both $v_{j'}$ and $v_{i'}$ block (v_j, v_i), where $v_{j'} = \arg\min_{v_b \in \text{chain}_{v_i}(v_{j+1}, v_{i-1})} \measuredangle_{v_i}(v_l, v_b)$ and $v_{i'} = \arg\min_{v_b \in \text{chain}_{v_j}(v_{j+1}, v_{i-1})} \measuredangle_{v_j}(v_b, v_l)$. This is then also true in our original polygon $\mathcal{P}$ and thus $v_{i'} \in \text{chain}(v_j, v_{j'})$ as otherwise $v_{j'}$ would block $(v_j, v_{i'})$ and $v_{i'}$ would block $(v_{j'}, v_i)$ contradicting the definition of $v_{j'}$ and $v_{i'}$, respectively. Observe that $v_{i'}$ is the last vertex in $\text{chain}(v_{i+1}, v_i)$ visible to v_j and $v_{j'}$ is the first vertex in $\text{chain}(v_j, v_{j-1})$ visible to v_i.

By applying Lemma 3 to $\mathcal{P}'$, we know that both $v_{j'}$ and $v_{i'}$ are left of the oriented line $\overline{v_j v_i}$. This means $\measuredangle^{\uparrow}_{v_i}(v_l, v_j) = \measuredangle_{v_i}(v_l, v_{j'}) < \measuredangle_{v_i}(v_l, v_j)$ and $\measuredangle^{\uparrow}_{v_j}(v_i, v_l) = \measuredangle_{v_j}(v_{i'}, v_l) < \measuredangle_{v_j}(v_i, v_l)$ and thus $\measuredangle^{\uparrow}_{v_i}(v_l, v_j) + \measuredangle^{\uparrow}_{v_j}(v_i, v_l) + \measuredangle_{v_l}(v_j, v_i) < \measuredangle_{v_i}(v_l, v_j) + \measuredangle_{v_j}(v_i, v_l) + \measuredangle_{v_l}(v_j, v_i) = \pi$, which is a contradiction with our assumption that v_l is a triangle witness of (v_i, v_j). □

Theorem 1. *The triangle witness algorithm is correct, computes a unique solution, and can be implemented with a running time of* $O(n^3 \log n)$.

Proof. As the edges in E_{vis} are the same as the edges stored in F and the same as the edges stored in B throughout the algorithm, it is sufficient to show that after step k of the iteration both F and B contain exactly the edges between vertices that are at most k steps apart along the boundary. As no two vertices can be further apart than $\lceil \frac{n}{2} \rceil$ steps along the boundary, this immediately implies that

E_{vis} eventually contains exactly the edges of the visibility graph. More precisely, we inductively show that after step k of the iteration, $F[v_i]$ contains the vertices of $\text{chain}_{v_i}(v_{i+1}, v_{i+k})$ and $B[v_i]$ contains the vertices of $\text{chain}_{v_i}(v_{i-k}, v_{i-1})$ for all $v_i \in V$. For sake sake of simplicity we write $F[v_i] = \text{chain}_{v_i}(v_{i+1}, v_{i+k})$ and $B[v_i] = \text{chain}_{v_i}(v_{i-k}, v_{i-1})$.

The discussion for $k = 1$ is trivial as every vertex has an edge to both its neighbors. The algorithm initializes F and B to consist of these edges. It remains to show for all $0 \leq i < n$ that assuming $F[v_i] = \text{chain}_{v_i}(v_{i+1}, v_{i+k-1})$ and $B[v_i] = \text{chain}_{v_i}(v_{i-k+1}, v_{i-1})$ after step $k-1$, we have $F[v_i] = \text{chain}_{v_i}(v_{i+1}, v_{i+k})$ and $B[v_i] = \text{chain}_{v_i}(v_{i-k}, v_{i-1})$ after step k.

The algorithm adds an edge between two vertices v_i and v_{i+k}, if and only if there is a vertex $v_l \in \text{chain}(v_{i+1}, v_{i+k-1})$ with $v_l \in F[v_i] \wedge v_l \in B[v_{i+k}]$ for which $\alpha_i + \alpha_j + \alpha_l = \pi$, where $\alpha_i, \alpha_j, \alpha_l$ are defined as in Algorithm 1. As v_i and v_l are less than k steps apart on the boundary, the induction assumption implies that $F[v_i] = \text{chain}_{v_i}(v_{i+1}, v_{i+k-1})$ and $B[v_{i+k}] = \text{chain}_{v_{i+k}}(v_{i+1}, v_{i+k-1})$. Thus, $v_l \in F[v_i] \wedge v_l \in B[v_{i+k}]$ is equivalent to $\{v_i, v_l\}, \{v_{i+k}, v_l\} \in E_{\text{vis}}$ and by Lemma 4 it suffices to show that $\alpha_i = \measuredangle^{\uparrow}_{v_i}(v_l, v_{i+k}), \alpha_j = \measuredangle^{\uparrow}_{v_{i+k}}(v_i, v_l), \alpha_l = \measuredangle_{v_l}(v_{i+k}, v_i)$ for all $v_l \in F[v_i] \cap B[v_{i+k}]$. Again by induction $F[v_i] = \text{chain}_{v_i}(v_{i+1}, v_{i+k-1})$ and thus $\text{vis}_{F[v_i][v_l]}(v_i) = v_l$ and $\text{vis}_{|F[v_i]|+1}(v_i) = \arg\min_{v_b \in \text{chain}_{v_i}(v_{i+k}, v_{i-1})} \measuredangle_{v_i}(v_{i+1}, v_b)$ and we thus get $\alpha_i = \angle_{v_i}(F[v_i][v_l], |F[v_i]| + 1) = \measuredangle^{\uparrow}_{v_i}(v_l, v_{i+k})$. Similarly as v_l and v_{i+k} are less than k steps apart on the boundary, we get $\alpha_j = \measuredangle^{\uparrow}_{v_{i+k}}(v_i, v_l)$. Further, with the induction assumption we also have $\text{vis}_{F[v_l][v_{i+k}]}(v_l) = v_{i+k}$ and $\text{vis}_{B[v_l][v_i]}(v_l) = v_i$ and thus $\alpha_l = \angle_{v_l}(F[v_l][v_j], B[v_l][v_i]) = \measuredangle_{v_l}(v_{i+k}, v_i)$.

The uniqueness of the algorithm's solution follows immediately from the fact that the existence of a triangle witness is necessary and sufficient for two vertices to see each other.

For every vertex v_i and every $k = 1, 2, \ldots, \lceil \frac{n}{2} \rceil$, the algorithm has to iterate over all candidates $v_l \in \text{chain}(v_{i+1}, v_{i+k-1})$ of a triangle witness of (v_i, v_{i+k}). In total at most $O(n^3)$ such combinations have to be examined. In order to decide whether a particular v_l is a triangle witness of (v_i, v_{i+k}), first the algorithm has to decide whether v_l is visible to both v_i and v_{i+k}. If we use a self-balancing tree data structure for $F[v_i]$ and $B[v_{i+k}]$ for all choices of i and k, this decision requires $O(\log n)$ time. Summing the corresponding angles and comparing the result to π takes constant time. Hence the total running time is $O(n^3 \log n)$. □

Note that as the triangle witness algorithm computes a unique solution, it provides an immediate way of identifying inconsistent input, i.e. angle data that does not belong to any polygon. If upon termination of the algorithm $|F[v_i] \cup B[v_i]| \neq d(v_i)$ for some vertex v_i, the input must be inconsistent. Otherwise, we can compute a triangulation of the visibility graph and infer the shape of it in the plane. Then the input was consistent if and only if the computed shape is valid (i.e., if this gives a simple polygon that is consistent with the input sequence of angle measurements).

References

1. Biedl, T., Durocher, S., Snoeyink, J.: Reconstructing polygons from scanner data. In: Dong, Y., Du, D.-Z., Ibarra, O. (eds.) ISAAC 2009. LNCS, vol. 5878, pp. 862–871. Springer, Heidelberg (2009)
2. Bilò, D., Disser, Y., Mihalák, M., Suri, S., Vicari, E., Widmayer, P.: Reconstructing visibility graphs with simple robots. In: Proceedings of the 16th International Colloquium on Structural Information and Communication Complexity, pp. 87–99 (2009)
3. Chalopin, J., Das, S., Disser, Y., Mihalák, M., Widmayer, P.: How simple robots benefit from looking back. In: Proceedings of the 7th International Conference on Algorithms and Complexity (to appear)
4. Formann, M., Woeginger, G.: On the reconstruction of simple polygons. Bulletin of the EATCS 40, 225–230 (1990)
5. Ghosh, S.K.: Visibility Algorithms in the Plane. Cambridge University Press, Cambridge (2007)
6. Ghosh, S.K., Goswami, P.P.: Unsolved problems in visibility graph theory. In: Proceedings of the India-Taiwan Conference on Discrete Mathematics, pp. 44–54 (2009)
7. Jackson, L., Wismath, K.: Orthogonal polygon reconstruction from stabbing information. Computational Geometry 23(1), 69–83 (2002)
8. Jansen, K., Woeginger, G.: The complexity of detecting crossingfree configurations in the plane. BIT Numerical Mathematics 33(4), 580–595 (1993)
9. Kameda, T., Yamashita, M.: On the reconstruction of polygons with (simple) robots. Personal communication (2009)
10. Rappaport, D.: On the complexity of computing orthogonal polygons from a set of points. Technical Report SOCS-86.9, McGill University, Montréal, Canada (1986)
11. Sidlesky, A., Barequet, G., Gotsman, C.: Polygon reconstruction from line cross-sections. In: Proceedings of the 18th Annual Canadian Conference on Computational Geometry, pp. 81–84 (2006)
12. Skiena, S., Smith, W., Lemke, P.: Reconstructing sets from interpoint distances. In: Proceedings of the Sixth Annual Symposium on Computational Geometry, pp. 332–339 (1990)
13. Snoeyink, J.: Cross-ratios and angles determine a polygon. In: Proceedings of the 14th Annual Symposium on Computational Geometry, pp. 49–57 (1998)
14. Suri, S., Vicari, E., Widmayer, P.: Simple robots with minimal sensing: From local visibility to global geometry. International Journal of Robotics Research 27(9), 1055–1067 (2008)