Reconstructing Polygons from Scanner Data

Therese Biedl[1,⋆], Stephane Durocher[2], and Jack Snoeyink[3]

[1] David R. Cheriton School of Computer Science, University of Waterloo
biedl@cs.uwaterloo.ca
[2] Department of Computer Science, University of Manitoba
durocher@cs.umanitoba.ca
[3] Department of Computer Science, University of North Carolina at Chapel Hill
snoeyink@cs.unc.edu

Abstract. A range-finding scanner can collect information about the shape of an (unknown) polygonal room in which it is placed. Suppose that a set of scanners returns not only a set of points, but also additional information, such as the normal to the plane when a scan beam detects a wall. We consider the problem of reconstructing the floor plan of a room from different types of scan data. In particular, we present algorithmic and hardness results for reconstructing two-dimensional polygons from points, point/normal pairs, and visibility polygons. The polygons may have restrictions on topology (e.g., to be simply connected) or geometry (e.g., to be orthogonal). We show that this reconstruction problem is NP-hard in most models, but for some assumptions allows polynomial-time reconstruction algorithms which we describe.

1 Introduction

Range scanners have been configured in many ways: looking in to capture objects on a platform or in-situ, looking down to capture terrain or urban environments, and looking out to capture rooms or factory floors. Some scanners [3,11] provide – in addition to point coordinates – surface labels, normals, or unobstructed portions of scanned rays.

The problem of reconstructing surfaces from the resulting point clouds has been given both theoretical and practical consideration. Theoretical solutions can provably reconstruct the correct surface when the sample points are sufficiently dense relative to local feature size [1,2]. Applied solutions handle noisy data and often incorporate additional information such as estimated normals [11].

We consider the problem of reconstructing the two-dimensional floor plan of a polygonal room using different types of scanned data. Specifically, we ask whether having additional information about each data point and knowing something about the geometry (monotonicity and/or orthogonality) of the room allows efficient reconstruction from less dense data.

⋆ Supported by NSERC.

Y. Dong, D.-Z. Du, and O. Ibarra (Eds.): ISAAC 2009, LNCS 5878, pp. 862–871, 2009.

1.1 Models and Problem Definition

We consider five models for input data that may be obtained by scanning a room with one or more scanners, as illustrated in Fig. 1.

1. A *point scan* is a set of points, each of which lies on a wall (edge) of the scanned room (polygon).
2. A *point-wall scan* is a point scan for which each point records the line containing the wall on which it lies (i.e., the orientation of the wall).
3. A *point-normal scan* is point-wall scan for which each point-line pair records a normal perpendicular to the line that points towards the room's interior.
4. A *segment scan* is a point-normal scan for which each point records the position of its scanner; this implies that the entire line segment from a scanner to the corresponding scan point must be inside the room.
5. A *visibility-polygon scan* is a set of visibility polygons, i.e., the entire region visible from each scanner.

Given n observations under of one of the five models, the polygon reconstruction problem is to determine whether there exists a polygon that is consistent with the input data. Without additional restrictions, the answer is always "yes" in the point-scan model, as well as in all five models if a solution may include additional edges not encountered by any input point. In this paper, therefore, we assume that each wall has been seen, i.e., that each polygon edge contains at least one scan point.

1.2 Related Work

We have already mentioned results [1,2,11] on reconstructing shapes from densely-spaced samples on the surface in the point-scan model. Because we primarily

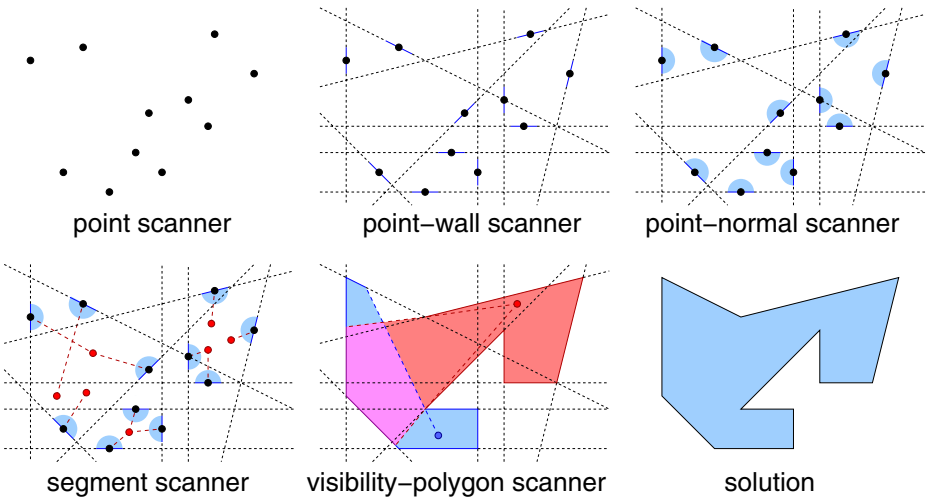

Fig. 1. Input instances of the five scan models and a common solution

consider scan models that provide additional information, sample points need not be closely spaced as long as each edge includes at least one sample point. Thus, our approaches are more closely related to previous work on reconstructing a polygon from a given vertex set. O'Rourke [12] gives an $O(n \log n)$ time algorithm for reconstructing an orthogonal polygon when all vertices must meet edges at a right angle; when a solution exists, it is unique. This problem is NP-hard if edges at a vertex meet either straight or at right angles [14] and also NP-hard if edges must be parallel to one of three (or more) given directions [5].

The reconstruction problem can be formulated as a matching problem with additional restrictions in a graph $G = (V, E)$. Each sample point corresponds to a segment on the polygon's boundary: let V contain two vertices for each sample point, one for each direction out from the segment. Join two vertices by an edge in E if the corresponding rays intersect. The polygon reconstruction problem reduces to finding a spanning subgraph $H \subseteq G$ that has specific properties. In particular, we require that H be a perfect matching that is simple (no matching edges cross each other). Furthermore, if each pair of vertices induced by a sample point is joined by an edge, the resulting subgraph must be connected.

If the constraints of simplicity and connectivity are dropped, the problem is reducible to finding a perfect matching and is solvable in polynomial time [4,6,7,10]. Adding either constraint renders the problem hard: finding a non-crossing 2-factor in a geometric graph was shown to be NP-hard by Jansen and Woeginger [9], and a connected 2-factor is simply a Hamiltonian cycle, which is well known to be NP-hard to find [6], even in grid graphs [8]. Neither of these results, however, directly implies hardness for the polygon reconstruction problems we consider.

1.3 Our Results

We first show that the reconstruction problem is NP-hard (Section 2). Our reduction is to the visibility-polygon scan model, but straightforward modifications can reduce to the point-wall, point-normal, or segment scan models.

For positive results, we consider geometric restrictions to the allowable configurations of polygons. A *star-shaped polygon* is entirely visible from some point in its interior (i.e., there is a single scanner). The interior of a *monotone*[1] *polygon* intersects every vertical line in at most one line segment. The boundary of a monotone polygon splits into two chains, the *upper* and *lower chains*, both of which are monotone. Every edge in an *orthogonal polygon* is either horizontal or vertical.

Our hardness result implies that the reconstruction problem remains NP-hard even for orthogonal polygons. We show that two cases for monotone polygons can be solved in polynomial time: orthogonal monotone polygons for point-wall scans (Section 3), and monotone polygons for point-normal scans (Section 4). The reconstruction problem is also solvable for star-shaped polygons (Section 5). Finally, we present a lower bound showing that the running times of our algorithms are optimal (Section 6).

[1] For simplicity, we use the term *monotonicity* to refer to *x-monotonicity*.

2 Hardness Results

In this section, we prove that reconstructing a simply-connected polygon from a visibility-polygon scan is NP-hard. We use a reduction from ORTHOGONAL NON-CROSSING SPANNING TREE, which was shown to be NP-hard by Jansen and Woeginger [9]. An *orthogonal graph* is a graph drawn in the plane such that every edge is either a horizontal or vertical line segment connecting two vertices and no edge contains any vertex in its interior.

ORTHOGONAL NON-CROSSING SPANNING TREE
Instance. An orthogonal graph G.
Question. Find a graph $H \subseteq G$ that is a spanning tree of G such that no two edges in H cross.

Theorem 1. *Polygon reconstruction under the visibility-polygon scan model is NP-hard.*

Proof. Given any orthogonal graph G, we construct an instance of the visibility-polygon scan problem, $f(G)$, by replacing each vertex v with the vertex gadget $f(v)$ illustrated in Fig. 2. In this gadget, there is a gap in the corresponding polygon edge for every neighbour of the vertex. This allows either connecting to the corresponding neighbouring vertex gadget via the corridor formed by a pair of parallel edges (blue, dashed), or closing off the gap by extending an edge (red, dotted). If a vertex has degree less than four, then the positions of the corresponding scanners can be moved accordingly such that there is no gap (Fig. 2B).

Let d denote the minimum Euclidean distance between the position of any two vertices in the orthogonal drawing of G. Choose any $\epsilon \in (0, d/2)$. Now $f(G)$ consists of a set of vertex gadgets such that a gadget of width and height $d/2 - \epsilon$ is centered at the position of every vertex $v \in V$. See Figs. 2 and 3.

Components $f(v_1)$ and $f(v_2)$ can be joined by a pair of horizontal or vertical parallel edges forming a corridor if and only if vertices v_1 and v_2 are adjacent in G. Each edge in the corridor completes a partial edge in one of the two vertex gadgets. Note that the resulting instance $f(G)$ can be constructed in time proportional to the size of G on a Cartesian grid.

If G has a non-crossing spanning tree, then $f(G)$ has a simple polygonal solution formed by including the corridors that correspond to edges of the spanning tree. On the other hand, if $f(G)$ has a simple polygonal solution, then all vertex gadgets must be joined. Since every edge of the polygon must be seen by a scan, joining edges complete partial edges, and are therefore orthogonal. Since the polygon must be simple, the solution selects both or neither edge in a corridor, and edges from two crossing corridors cannot be selected simultaneously. Therefore, G has a non-crossing spanning tree. □

Since all edges in the reduction are orthogonal, the visibility-polygon scan problem remains NP-hard for orthogonal polygons. The construction can be modified to show hardness for the point-normal scan and segment scan models.

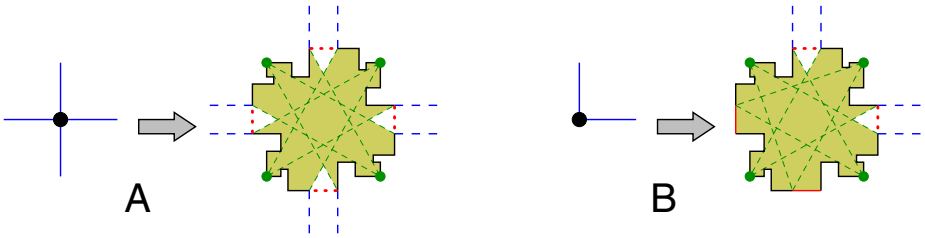

Fig. 2. The vertex gadget is a portion of the polygon with four scanners: vertices of degree four (**A**) and degree two (**B**). Dashes indicate how a gadget may be closed or continued, provided it matches a corresponding gadget on the other end. Graph edge crossings that are not vertices need no gadget.

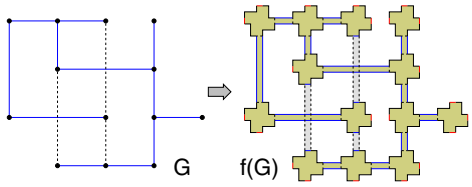

Fig. 3. A spanning tree of a graph G and the corresponding simple polygon in $f(G)$

3 Orthogonal Monotone Polygons

Since the general reconstruction problem is NP-hard, we consider special cases that are solvable in polynomial time. Many actual room layouts are orthogonal and monotone, motivating our consideration of these natural geometric constraints in this section. We show that these can be reconstructed uniquely from point-wall scans, i.e., each data point returns whether its edge is horizontal (H) or vertical (V).

Theorem 2. *A monotone orthogonal polygon can be reconstructed from a point-wall scan in $O(n \log n)$ time. Moreover, the solution is unique.*

Initially, let us assume that no edge contains two data points and that no data point is at a vertex; we describe later how to resolve such instances. We represent the input as a sequence σ of symbols over the alphabet $\{V, H\}$ in left-to-right order (breaking ties arbitrarily) in correspondence to whether the associated edge is vertical or horizontal. It will suffice to determine for each symbol whether it belongs to the lower or the upper chain. Our approach is to begin at a subsequence of σ for which the solution is uniquely determined locally, and then to propagate the solution, first to the right and then to the left.

Sequence σ must begin and end with V for the leftmost and rightmost edges. Under our assumptions, σ has equally many Vs and Hs, so it contains a subsequence HH – two horizontal edges with no vertical edge between them. The one

with larger y-coordinate must be in the upper chain and the other in the lower chain. (They cannot have the same y-coordinate since otherwise they would belong to the same edge.)

Now we extend the chains rightwards. The next element of σ cannot be H, otherwise some horizontal edge would contain two data points. Therefore the next element must be V; let p_v denote the corresponding data point. If the next element of σ is H (with corresponding data point denoted p_h), then the chains can be expanded as follows:

- If p_v is above the upper chain, then it belongs to the upper chain (see Fig. 4A).
- If p_v is below the lower chain, then it belongs to the lower chain (see Fig. 4B).
- If p_v is strictly between the two chains and p_v is above p_h, then it belongs to the upper chain (see Fig. 4C).
- If p_v is strictly between the two chains and p_v is below p_h, then it belongs to the lower chain (see Fig. 4D).
- Note that p_v cannot be at the same height as one of the chains, otherwise there would be a crossing or a data point at a vertex.
- In all cases, p_h belongs to the same chain as p_v.

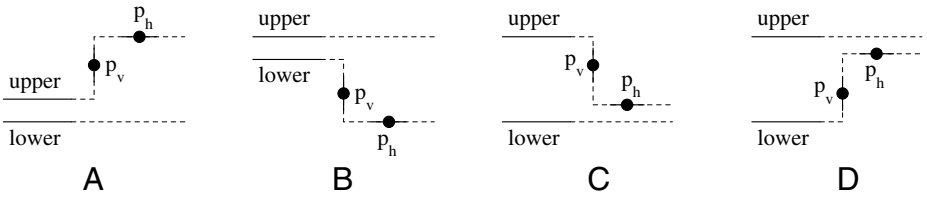

Fig. 4. Four cases for resolving a substring VH

Thus, if the next two elements of σ are VH, we can resolve these two edges. Moreover, again we know the y-coordinates of the last horizontal edge of the upper and lower chains, and hence can repeat until the next two elements of σ are VV. Here we can determine similarly to which chains the vertical edges belong, but then are stopped since no y-coordinates of the upper and lower chains are known to the right.

For every substring VV, there must be a substring HH later on since σ ends with V. We jump forward to this occurrence of HH, and then resolve rightward from then on until we reach another VV, jump forward to the next HH, and so on, until we have reached the rightmost data point.

We then repeat the same process in the opposite direction: The first scan resolves all intervals of the form HH to VV or rightmost V, and the second resolves from VV or leftmost V to HH. The time complexity of the algorithm is linear once the data points are sorted by x-coordinates. The resulting orthogonal polygon is unique since each edge is deterministically assigned to a chain.

To remove the initial assumptions: multiple data points on a horizontal edge can be detected and omitted during the above propagation since the points

have identical y-coordinates. Detecting multiple data points on a vertical edge is slightly more complicated. We can deduce the position of the substring VV by the absence of the substring HH due to multiple data points; we then delete one of the data points without affecting the solution. Finally, vertices can report VH or HV as they choose; if we find the sequence begins or ends with H, we assign responsibility to a vertex and swap with the adjacent V. □

4 Monotone Polygons

In this section we consider the reconstruction problem for monotone (not necessarily orthogonal) polygons from a point-normal scan. In this case, each input point knows the orientation and interior of the polygon boundary passing through it. In the monotone setting, these half-spaces determine whether each non-vertical edge belongs to the upper or lower chain of the polygon. This leaves the set of vertical edges to be assigned to chains. Nevertheless, the problem is non-trivial; in particular, a solution is not necessarily unique (e.g., see Fig. 5).

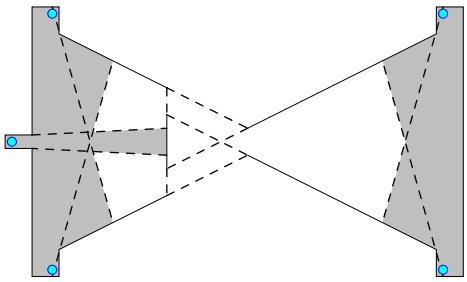

Fig. 5. Even under the visibility-polygon scan model this input instance has two distinct monotone solution polygons; the central vertical edge could belong to the lower or upper chain

Theorem 3. *A monotone polygon can be reconstructed from a point-normal scan in* $O(n \log n)$ *time.*

We sketch the ideas behind the dynamic programming algorithm that determines the chain to which vertical edges are assigned. Scan all data points from left to right and update a function that stores whether there is a partial solution (in the form of an upper and lower chain) up to the current x-coordinate, with some conditions on where the upper and lower chains end.

Let t denote an x-coordinate that is not the coordinate of any data point. The *upper-left line* of t is the line through the last data point before t for which the edge is not vertical and is in the upper chain (i.e., the associated half-space points downward). The *upper-right line* of t is the line through the first data point after t for which the edge is not vertical and is in the upper chain. We define the *lower-left* and *lower-right* lines of t analogously.

Observe that the vertical line through t intersects the upper chain of any solution necessarily in either the upper-left line or the upper-right line; otherwise one of the corresponding data points could not be used for the upper chain (and could not be used for the lower chain by the given normals). We compute a partial solution and prescribe which of the two lines it uses for the upper chain, and correspondingly for the lower. Thus, define $f(t, u, \ell) \in \{\text{true}, \text{false}\}$, where $u, \ell \in \{L, R\}$, and $f(t, u, \ell) = \text{true}$ if and only if there exist disjoint, monotone, upper and lower chains that may end at t with the upper line $u \in \{L, R\}$ and lower line $\ell \in \{L, R\}$, as in Fig. 6.

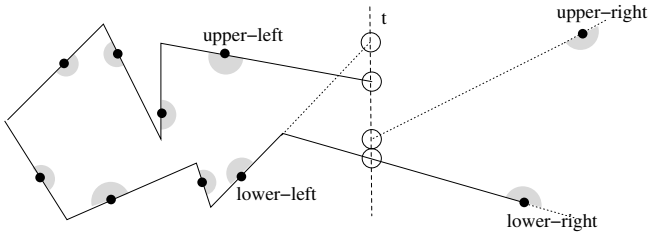

Fig. 6. In this example $f(t, L, R) = \text{true}$. White circles indicate points in which the upper/lower chain might be required to end.

We can initialize $f(t, u, \ell)$ at $t = -\infty$ and update $f(t, u, \ell)$ as t increases. We omit the details, most of which are straightforward; the full paper gives details on one of the more complex cases. Each update can be done in constant time. The time complexity for this algorithm is linear once the data points have been sorted by x-coordinate, resulting in a total running time of $O(n \log n)$. □

5 Star-Shaped Polygons

We briefly describe simple results for reconstructing a star-shaped polygon, i.e., a polygon that is entirely visible from some point in its interior. The region of points that sees all of a star-shaped polygon is its *kernel*.

Reconstructing a star-shaped polygon is straightforward in the point-normal model. Any point in the kernel must be visible to all data points. To compute the kernel, it suffices to compute the intersection I of the set of half-spaces associated with data points; this can be achieved in $O(n \log n)$ time [13]. Either all or none of the points in I are visible to all data points. Thus, to compute the polygon, select any point o in the interior of I, sort all data points in clockwise order around o, and compute the polygon defined by them in this order; either this polygon is star-shaped or there is no solution. Consequently, a solution is unique if it exists.

Theorem 4. *A star-shaped polygon can be reconstructed from a point-normal scan in $O(n \log n)$ time. Moreover, the solution is unique.*

We can also reconstruct a star-shaped polygon from a point-wall scan, but the time complexity increases. Consider the arrangement defined by the set of lines that pass through walls. As for point-normal scans, the kernel of a star-shaped polygon must be one of the cells defined by this arrangement, i.e., one of the maximal connected regions that do not contain a point on a line. There are $O(n^2)$ such cells for n lines. For each cell we can select a point o and attempt to reconstruct a star-shaped polygon with o in its kernel as explained above. The corresponding time complexity is $O(n^3 \log n)$. We suspect that this time can be improved: instead of repeating the $O(n \log n)$ test in every cell, it might be possible to update the intersection of half-planes dynamically each time a half-plane is crossed. Furthermore, we believe that the solution, if one exists, is unique. Both of these questions remain open.

Theorem 5. *A star-shaped polygon can be reconstructed from a point-wall scan in polynomial time.*

6 Lower Bound

We show the following lower bound which follows by a reduction from sorting. Details are omitted due to space constraints.

Theorem 6. *Any algorithm that reconstructs an orthogonal polygon from point-scans, point-wall scans, point-normal scans, or segment scans requires $\Omega(n \log n)$ comparisons. Furthermore, this lower bound also applies to the cases for which a solution must be orthogonal monotone or orthogonal, monotone, and star-shaped.*

7 Discussion and Directions for Future Research

If a solution is not unique, a natural question is to determine the number of additional scanners necessary to reveal the true solution. This question is NP-hard since Theorem 1 shows hardness for an instance of the corresponding decision problem. Approximation algorithms might be interesting to consider.

Several variants of our problem have not yet been considered. In particular:

- Can we reconstruct a monotone polygon from a point-wall scan?
- What other restrictions on a solution make reconstruction feasible in polynomial time? For example, a reasonable assumption could be that a room has four walls, each of which is a polygonal chain: two x-monotone walls and two y-monotone walls.
- Finally, a natural question is to consider the corresponding problems in three or higher dimensions.

Acknowledgements

The authors wish to thank Joseph Mitchell for suggesting that we examine lower bounds (Section 6) as well as a possible technique for improving the running time of our algorithm for star-shaped polygons under the point-wall scan model (Section 5).

References

1. Amenda, N., Choi, S., Kolluri, R.: The power crust, union of balls and the medial axis. Comp. Geom.: Theory & App. 19(2–3), 127–153 (2001)
2. Amenta, N., Choi, S., Kolluri, R.: The power crust. In: Proc. ACM Symp. Solid Model. & App., pp. 249–260 (2001)
3. DeltaSphere, Inc. Deltasphere 3D laser scanner (2001), `http://www.deltasphere.com/DeltaSphere-3000.html`
4. Edmonds, J.: Paths, trees, and flowers. Can. J. Math. 17, 449–467 (1965)
5. Formann, M., Woeginger, G.J.: On the reconstruction of simple polygons. Bull. Eur. Assoc. Theor. Comp. Sc. 40, 225–230 (1990)
6. Garey, M.R., Johnson, D.S.: Computers and Intractability. W.H. Freeman, New York (1979)
7. Hopcroft, J.E., Karp, R.M.: An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. SIAM J. Comp. 1(4), 472–484 (1988)
8. Itai, A., Papadimitriou, C.H., Szwarcfiter, J.L.: Hamilton paths in grid graphs. SIAM J. Comp. 11(4), 676–686 (1982)
9. Jansen, K., Woeginger, G.J.: The complexity of detecting crossingfree configurations in the plane. BIT 33(4), 580–595 (1993)
10. Lovász, L., Plummer, M.D.: Matching theory. In: Ann. Disc. Math. North-Holland Mathematics Studies, vol. 29 (1986)
11. Nyland, L., Lastra, A., McAllister, D., Popescu, V., McCue, C.: Capturing, processing and rendering real-world scenes. In: El-Hakim, S.F., Gruen, A. (eds.) Videometrics and Optical Methods for 3D Shape Measurement, Electronic Imaging. SPIE Int. Soc. Optical. Eng. (2001)
12. O'Rourke, J.: Uniqueness of orthogonal connect-the-dots. In: Toussaint, G.T. (ed.) Computational Morphology, pp. 97–104. Elsevier, Amsterdam (1988)
13. Preparata, F., Shamos, M.: Computational Geometry: An Introduction. Springer, Heidelberg (1985)
14. Rappaport, D.: On the complexity of computing orthogonal polygons from a set of points. Technical Report SOCS-86.9, McGill University, Montréal, Canada (1986)