Computational Geometry 44 (2011) 409–417

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Reconstructing orthogonal polyhedra from putative vertex sets ☆

Therese Biedl [b], Burkay Genç [a,*]

[a] Faculty of Engineering and Computer Sciences, Izmir University of Economics, Sakarya Cad. No: 156, Balcova, Izmir, Turkey
[b] David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada

ARTICLE INFO

Article history:
Received 22 August 2008
Accepted 19 April 2011
Available online 22 April 2011
Communicated by G. Toussaint

Keywords:
Reconstruction
Vertex set
Orthogonal polyhedra

ABSTRACT

In this paper we study the problem of reconstructing orthogonal polyhedra from a putative vertex set, i.e., we are given a set of points and want to find an orthogonal polyhedron for which this is the set of vertices. This is well-studied in 2D; we mostly focus on 3D, and on the case where the given set of points may be rotated beforehand. We obtain fast algorithms for reconstruction in the case where the answer must be orthogonally convex.

1. Introduction

In this paper, we study the following problem: Given a set of points S in 3D, is there an orthogonal polyhedron for which the set of vertices is exactly S? And if there is such a polyhedron, is it unique? (Precise definitions will be given in Section 2.)

1.1. Motivation and our results

The problems studied in this paper are related to pattern and object recognition from point sets [1–3]. Given a point set in three dimensions, it is possible to reconstruct a convex object that includes these points by finding the convex hull of the points in the set. However, reconstructing a non-convex object from a point set is quite difficult and the resulting objects are not necessarily unique. It is known that even reconstruction of orthogonal shapes from points in two dimensions is a difficult problem [4]. Does it help to know that the given points are the vertices of the object to be reconstructed? This is almost like the childhood game "connect-the-dots", except that the points are not numbered. O'Rourke [5] provides an algorithm for orthogonal polygons if the given point set is the exact set of vertices. Is it possible to extend this result to three dimensions for orthogonal polyhedra? What if the points correspond to a "rotated" orthogonal polygon or polyhedron?

The second interest is succinct data structures for storing polyhedra. The "standard" boundary representation (storing vertex coordinates and the graph of the polyhedron as doubly-connected edge list; see e.g. [6]) contains much redundancy when applied to an orthogonal polyhedron. What information can be omitted while maintaining uniqueness of the polyhedron? This papers studies the *vertex representation*, where we store only the vertex coordinates, and the problem can be re-phrased as follows: Is the vertex representation unambiguous, i.e., does it give rise to one unique orthogonal polyhedron?

☆ Research supported by NSERC.
* Corresponding author.
E-mail addresses: biedl@uwaterloo.ca (T. Biedl), burkay.genc@ieu.edu.tr (B. Genç).

0925-7721/$ – see front matter
doi:10.1016/j.comgeo.2011.04.002

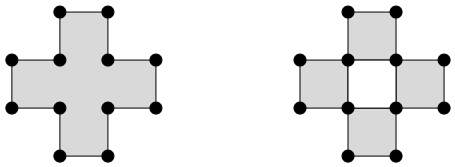

Fig. 1. Two polygons (one touching itself) that have the same set of vertices. From [7].

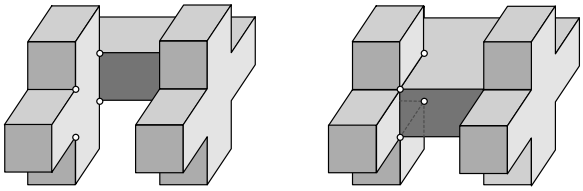

Fig. 2. Two orthogonal polyhedra that have the same vertex set.

Bournez et al. [7] gave an example where this representation is ambiguous, even in 2D (see Fig. 1). In this example, the polygon boundary touches itself repeatedly. If (as is more common) this is not allowed, then O'Rourke's algorithm [5] tests whether a set of 2D points belongs to an orthogonal polygon.

In 3D the vertex representation is indeed ambiguous, see Fig. 2. But this example is not orthogonally convex, so what is the situation if we restrict ourselves to orthogonally convex polyhedra? The precise problem studied in this paper is as follows:

Problem 1. Given a set S of n points in 3D, is there an orthogonally convex polyhedron for which the set of vertices is exactly S?

We give an $O(n \log n)$ algorithm to solve this problem. Also, there is a unique polyhedron, since the reconstructed polyhedron is in fact the orthogonally convex hull of P. We then study a generalization where we allow rotations, which is significantly more difficult.

Problem 2. Given a set of n points S in 3D, is there a rotation S' of S and an orthogonally convex polyhedron P such that the vertices of P are exactly S'?

It is quite easy to show that this problem is solvable in $O(n^3 \log n)$ time. We work on reducing this time complexity. First, we consider the 2D problem, and provide an $O(n \log n)$ algorithm to find a suitable polygon. If such a polygon exists, then at most one rotation can yield one (and the polygon is unique). Then we use this to reduce the time complexity for 3D to $O(n^2 \log n)$.

1.2. Related results

Reconstructing polygons from a given point set was first proposed by Steinhaus [8]. Unless the points are collinear, there always exists a polygon, and it can be found in many ways; Hurtado et al. [9] give a good overview on this topic. The three-dimensional version (reconstruct a general polyhedron from a set of points) appears to first have been solved by Grünbaum [10]; Hurtado et al. [9] mention many extensions and generalizations.

Noone appears to have studied reconstructing orthogonal polygons and polyhedra from point sets, but related results appear in the literature on representations of orthogonal polyhedra via their vertices. Aguilera and Ayala [11,12] showed how to reconstruct an orthogonal polyhedron from its vertices if we know which vertices are "extreme" (have degree 3). From this information they can generate all edges of the polyhedra. Bournez, Maler and Pnueli [7] assume that with each vertex we know whether a (pre-determined) octant is inside the polyhedron or outside. They do not reconstruct the edges or faces, but give a test that determines in $O(n \log n)$ time whether a given point is inside the polyhedron or outside. Contrasting our results with these papers, we demand less information (nothing except vertex coordinates), but in exchange can reconstruct only orthogonally convex polyhedra. Furthermore, we also consider rotations of the point set.

2. Definitions

A *polygon* is a set P in a plane whose boundary ∂P is a simple closed curve that consists of a finite number of line segments. A *vertex* of a polygon P is a point on its boundary where ∂P changes slope. An *edge* of a polygon is a line segment on its boundary that connects two vertices.

A set $S \subseteq R^3$ is *convex* if for any two points in S the line segment between them is also in S. The convex hull $CH(S)$ of a set S is the intersection of all convex sets that contain S.

An *orthogonal polygon* (sometimes also called *rectilinear polygon*) is a polygon whose boundary is composed entirely of axis-parallel segments. An orthogonal polygon P is *orthogonally convex* if any axis-parallel line intersects P in at most one line segment.

A *polyhedron* is a subset of R^3 whose boundary is a 2-manifold composed of finitely many interior-disjoint polygons. A *face* of a polyhedron is a maximal interior-connected planar region of the boundary of a polyhedron. Note that a face need not be a polygon, since its boundary may touch itself or have multiple components. A *vertex* of a polyhedron is a point on the boundary incident to at least three faces. An *edge* of the polyhedron is a maximal line segment on the boundary of a face that does not contain a vertex in its interior.

An *orthogonal polyhedron* is a polyhedron whose boundary is composed entirely of polygons that are lying within an *orthogonal plane*, i.e., a plane whose normal is one of the coordinate axes. A set is *orthogonally convex* if every intersection with an orthogonal plane is either empty or a single orthogonally convex polygon. Slightly abusing notation, we use the term *orthogonally convex polyhedron* to mean a polyhedron that is orthogonally convex and also orthogonal. (The latter is not required from prior definitions, but we will not study non-orthogonal polyhedra that are orthogonally convex.)

An orthogonally convex hull of a point set may be defined in different ways which are not all equivalent (see [13] for details for 2D). We use the following definition. The *orthogonally convex hull* of a set S (denote $OCH(S)$) is the intersection of all orthogonally convex sets that contain S. In general $OCH(S)$ need not be a polygon/polyhedron (because it is not always connected), but if S is a polygon/polyhedron, then $OCH(S)$ is also a polygon/polyhedron [14].

3. Reconstructing without rotation

The 2D-reconstruction problem ("Given a set of points, find an orthogonal polygon for which this is the set of vertices") has been solved by O'Rourke [5]. By sorting points on any orthogonal line, and then adding an edge between every other pair of points, the polygon (if it exists) can be reconstructed in $O(n \log n)$ time and is unique.

For 3D, the problem becomes harder. There may not be a unique answer as shown by O'Rourke [5]. In his example one answer is not actually a polyhedron because it is disconnected; our Fig. 2 shows another example where both answers are polyhedra. The complexity of finding some orthogonal polyhedron if one exists remains open. We study here orthogonally convex polyhedra, and show how to test whether a set of points is the vertex set of an orthogonally convex polyhedron in $O(n \log n)$ time.

So assume we are given a set of points $S \subseteq R^3$. One can easily see that if any orthogonally convex polyhedron P has vertex set S, then P must be the orthogonally convex hull of S (and in particular, is unique). However, there appears to be little work done on efficiently computing the orthogonally convex hull of a point set in three dimensions.

Computing the maxima is the closest result in the literature. We say that point q *dominates* point p if each coordinate of q is larger than the corresponding coordinate of p. A *maximum* of a point set S is then a point $p \in S$ that is not dominated by any point $q \in S$. The set of maxima of n points can be computed in $O(n \log n)$ time [15]. One can show easily that every vertex of the orthogonally convex hull is a maximum in one of the eight possible directions defined by reversing the dominance definition separately for each dimension, and vice versa, every maximal point of a point set S is on the orthogonally convex hull of S.

However, some obstacles remain. While computing maxima will detect whether S has any additional points in the interior of the orthogonally convex hull, it will not check that all vertices of the orthogonally convex hull are indeed in S. To do so, we would have to reconstruct all faces of the orthogonally convex hull, and we are not aware of an algorithm in the literature to do so.

We hence use a different approach to the problem of reconstructing an orthogonally convex polyhedron that does not use the idea of computing maxima. Our algorithm requires $O(n \log n)$ time and $O(n)$ space, where n is the number of points in the input set. It can be outlined in two steps: In the first step, sweep through all six orthogonal directions and construct the *shadows* (to be defined precisely later) of parts of the polyhedron. In the second step, reconstruct the edges and faces of the polyhedron from these shadows.

To explain the first step in more detail, the following definitions are necessary. Let P be an orthogonally convex polyhedron. Let $x_1 < x_2 < \cdots < x_t$ be the values for which some vertex of P has x-coordinate x_i. Each of these x_i determines a vertex *layer* of the polyhedron, which is the set of all vertices V_i with x-coordinate x_i. Let F_i be the x^--faces with x-coordinate x_i, where an x^-*-face* is a face with outward normal $(-1, 0, 0)$. Let π_i (the ith *shadow*) be the union of the faces of $F_1 \cup \cdots \cup F_i$ projected onto an x-plane (i.e., a plane perpendicular to the x-axis; y-planes and z-planes are defined analogously). One can easily see that π_i is the same as the projection of the polyhedron $P_i = P \cap \{x \leqslant (x_i + x_{i+1})/2\}$ onto an x-plane; see Fig. 3. In what follows, we will use *projection* to mean *projection onto an x-plane*; no other projections will be studied.

The crucial insight is that π_i can be computed from the V_i's alone, and from this the faces in F_i can be reconstructed. We need the following lemmas:

Lemma 1. *Let P be an orthogonally convex polyhedron. Then each shadow $\pi_i, i = 1, \ldots, t$, is an orthogonally convex polygon. Moreover, π_i is the orthogonally convex hull of the projections of $V_1, \ldots, V_i$.*

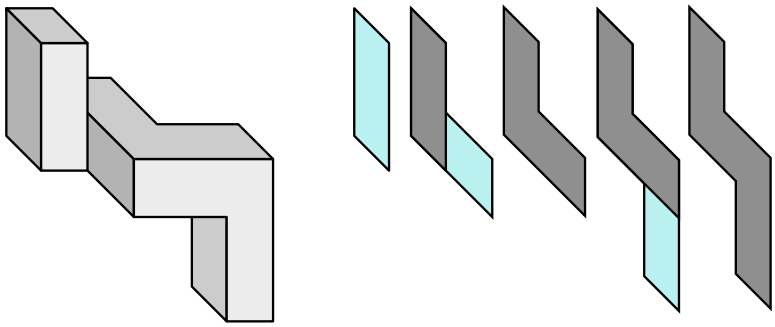

Fig. 3. An orthogonally convex polyhedron and its shadows.

Proof. Let ℓ be a horizontal or vertical line that intersects π_i. This line is the projection of a y-plane or z-plane that intersects P_i. The intersection of this plane with P_i is a connected set, since P (and hence P_i) is orthogonally convex. Therefore, $\ell \cap \pi_i$ (which is the projection of this intersection) is connected, i.e., a single line segment. So π_i is orthogonally convex.

Now let p be a point of π_i. Some x^--face f of P_i must contain a point that projects to p by definition of π_i. Face f is a subset of its orthogonally convex hull, so since p is in the projection of f, it is also in the orthogonally convex hull of the projection of the vertices of f. These vertices belong to some V_j, $j \leqslant i$, so p is in the orthogonally convex hull of the projection of $V_1 \cup \cdots \cup V_i$.

To show the other direction, observe that the projection of any vertex of $V_1 \cup \cdots \cup V_i$ belongs to π_i. Since π_i is orthogonally convex, it therefore is an orthogonally convex set containing all projections of these vertices, and as such, a superset of their orthogonally convex hull. □

Lemma 2. *Let U be the union of all faces in F_i, projected onto an x-plane. Then U is the closure of $\pi_i - \pi_{i-1}$.*

Proof. Directly from the definition of π_i, we have $\pi_i = \pi_{i-1} \cup U$, so all that remains to show is that $U \cap \pi_{i-1}$ is empty. This actually is not always true, but we show that $U \cap \pi_{i-1}$ contains no points in its interior, which implies the result since we take the closure.

So assume for contradiction that $U \cap \pi_{i-1}$ contains a point in its interior. Then a small open region around this point is also in $U \cap \pi_{i-1}$, and in it we can find a point p such that the x-line ℓ through p (i.e., the line parallel to the x-axis) does not intersect any edge or vertex of P. It intersects at least one x^--face among $F_1, \ldots, F_{i-1}$ because p belongs to π_{i-1}. It intersects another x^--face in F_i because p belongs to U. But then line ℓ intersects P in at least two line segments, because it intersects these x^--faces in their interior. This contradicts that P is an orthogonally convex polyhedron. □

These lemmas naturally imply an algorithm to compute the x^--faces: Sort all points with respect to their x-coordinates to obtain layers $V_1, \ldots, V_t$. For $i = 1, \ldots, t$, reconstruct the shadow π_i by computing the orthogonally convex hull of the projections of $V_1 \cup \cdots \cup V_i$. Then, compute the closure of $\pi_i - \pi_{i-1}$ and extract the maximal interior-connected regions, which are the x^--faces in F_i.

Repeating this for all six directions obtains the only possible set of faces that could belong to an orthogonally convex polyhedron P with vertex set S, and we can then easily check whether the set of vertices defined by these faces is exactly S.

This algorithm can easily be implemented in $O(n^2 \log n)$ time, if each π_i is reconstructed using the two-dimensional orthogonally convex hull algorithm by Ottmann et al. [13], which takes $O(n \log n)$ time and is applied $O(n)$ times (for each layer in each of six directions).

The time complexity can be improved by dynamically updating the orthogonal convex hull as new points are added at each layer, and immediately reconstructing the faces. Maintaining the orthogonally convex hull in 2D under additions of points can be done in $O(\log n)$ time per point. (This is well-known for general convex hulls, see e.g. [6], and can easily be modified to handle orthogonally convex hulls by maintaining four staircases that describe the orthogonally convex hull.)

To reconstruct the x^--faces in F_i quickly, observe that each connected components of $\pi_i - \pi_{i-1}$ is incident to at least one point in layer i. Hence, exploring the boundary of π_i from points in layer i, one can walk around each connected piece of $\pi_i - \pi_{i-1}$, hence find each face and explicitly list all vertices and edges. The time to do this is proportional to the number of vertices in the faces found; over all faces and all layers this is $O(n)$ time. Once the ith layer is swept, π_{i-1} is no longer necessary and can be discarded, hence the space complexity is $O(n)$.

Theorem 3. *The edges and faces of an orthogonally convex polyhedron P can be reconstructed from its vertex coordinates in $O(n \log n)$ time.*

Note that this algorithm assumes that the input set of vertices is the exact set of vertices of the orthogonally convex polyhedron. Therefore, it cannot be used as a means of computing the orthogonally convex hull of an arbitrary point set, though whether it could be modified to do so is an interesting open problem.

4. Rotated point sets

In this section, we study Problem 2, i.e., we allow a rotation to be applied to the point set before searching for a polyhedron. The following notation will be helpful: An *α-orthogonal polygon* (for $0 \leqslant \alpha < \pi/2$) is a polygon for which all edges have slope $\tan(\alpha)$ or $-\cot(\alpha)$. A *rotated orthogonal polygon* is an α-orthogonal polygon for some $0 \leqslant \alpha < \pi/2$. A *rotated orthogonal polyhedron* is a polyhedron obtained by applying some rotation matrix to an orthogonal polyhedron. Similarly we define *α-orthogonally convex polygon* and *rotated orthogonally convex polygon/polyhedron*.

The operations involved in the following sections require multiplications of real numbers. Therefore, all operations are assumed to be done under the real RAM model of computing.

4.1. Point sets in 2D

Problem 2 is far from trivial even in 2D, where it becomes the following: Given a set of points S in 2D, is there a rotated orthogonal polygon whose vertices are S?

A straightforward approach consists of trying all possible angles α for rotation, and for each of them, running the algorithm of O'Rourke [5]. To find the possible angles, compute the (conventional) convex hull $CH(S)$ of S. For each of the edges on the convex hull, try the rotation α that makes this edge horizontal or vertical. These rotations are the only possible candidates for obtaining a rotated orthogonal polygon because of the following easy fact:

Observation 4. Let P be an orthogonal polygon, and let $CH(P)$ be the convex hull of its vertices. Then there are at least four edges of P that are on the boundary of $CH(P)$. If P is orthogonally convex, then there are exactly four such edges (we will call them *extreme edges*), and they are edges of $CH(P)$.

The time complexity of this algorithm is $O(n^2 \log n)$ for an input set of n points, since the convex hull can be computed in $O(n \log n)$ time, gives at most n rotations, and for each of them O'Rourke's algorithm takes $O(n \log n)$ time.

One may wonder whether testing all rotations is really necessary. We posed this as an open problem at CCCG'07, and Löffler and Mumford soon thereafter proved that only one rotation can possibly yield a polygon (or for that matter, a connected graph) [16]. Moreover, they give an algorithm to find this rotation in $O(n^2)$ time. Applying O'Rourke's algorithm to this one rotation only hence decreases the running time of finding a rotated orthogonal polygon (if one exists) to $O(n^2)$. This also shows that the answer, if one exists, is unique.

Theorem 5. *The boundary of a rotated orthogonal polygon can be reconstructed from its vertex coordinates in $O(n^2)$ time.*

The time complexity can be improved even more for rotated orthogonally convex polygons, because the only possible rotation can be found faster. Hence the rest of this subsection deals with the following problem: Given a set S of points in R^2, construct a rotated orthogonally convex polygon with vertex set S if one exists. We solve this problem by proving that only one rotation can possibly work, a result that immediately follows from the work by Löffler and Mumford [16], but for which we give a separate proof that implies a simple fast algorithm to find it.

Given an edge e of a convex polygon C, let $H(e)$ be the closed half-disk of e (one half of the disk whose diameter is e) that intersects C. See also Fig. 4(a).

Lemma 6. *Let P be an α-orthogonally convex polygon for some $0 \leqslant \alpha < \pi/2$, and let $e = (v, w)$ be an edge of the convex hull $CH(P)$. If e is not an edge of P, then $H(e)$ contains at least one vertex $u \neq v, w$ of P.*

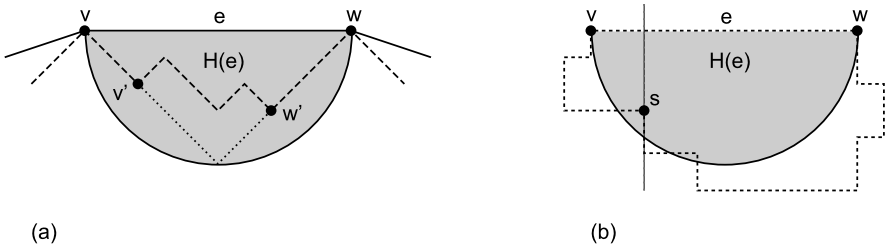

Fig. 4. (a) $H(e)$ must contain another vertex. In this picture, $\alpha \approx \pi/6$. Polygon P is dashed. (b) If e is the shortest extreme edge, then no vertex s can be in $H(e)$. Polygon P is dashed.

Proof. Assume e is not an edge of P. Vertex v has two incident edges in P, one of slope $\tan(\alpha)$ and the other of slope $-\cot(\alpha)$. Furthermore, these edges must be inside $CH(P)$, which (since the convex hull has angles less than π and the two edges are at angle $\pi/2$) means that one of them must enter $H(e)$. Call the other endpoint of this edge v'. Similarly one can argue that one incident edge of w must enter $H(e)$; call its other endpoint w'. See also Fig. 4(a).

Clearly, $v' \neq w$ and $w' \neq v$ since e is not an edge of P. If either one of v' and w' is inside $H(e)$, then we are done. But if both are outside $H(e)$, then the edges (v, v') and (w, w') cross, because the two corresponding rays have perpendicular slopes and hence meet exactly on $H(e)$. But the boundary of a polygon must not cross itself, so one vertex is inside $H(e)$. □

We need another observation. Let P be a rotated orthogonally convex polygon and let $e_0, \ldots, e_3$ be the four edges of P that are also edges of $CH(P)$, in order as encountered when walking along P (cf. Observation 4). These are the *extreme edges* of P.

Lemma 7. *Let P be a rotated orthogonal polygon and let $e = (v, w)$ be the shortest of its extreme edges. Then $H(e)$ contains no vertices other than v and w.*

Proof. After possible rotation, we may assume that e is horizontal with v left of w and P below e. See Fig. 4(b). Assume for contradiction that $s \neq v, w$ is a vertex inside $H(e)$ and consider the vertical line through s. This line intersects P at s and somewhere along edge (v, w). It cannot intersect the boundary of P anywhere else by orthogonal convexity. So it splits the boundary of P into two chains, one of which is monotone in y-direction since P is orthogonally convex. This monotone chain contains a vertical extreme edge, which cannot be longer than the distance from s to (v, w), which is at most $\|e\|/2$ since s is inside $H(e)$. Therefore e was not the shortest extreme edge, a contradiction. □

These two lemmas imply uniqueness of the rotation:

Theorem 8. *For a set S of points in plane, there exists at most one rotated orthogonally convex polygon whose set of vertices is exactly S.*

Proof. Assume for contradiction that S is the set of vertices of both an α-orthogonally convex polygon P and an α'-orthogonally convex polygon P' where $0 \leqslant \alpha \neq \alpha' < \pi/2$. By Lemma 7, there exists an extreme edge e of P for which $H(e)$ contains no other point of S. Edge e is on $CH(S) = CH(P) = CH(P')$, but it is not an edge of P', since P' has different slopes than P. So by Lemma 6, $H(e)$ contains some vertex of P', which is a point of S, a contradiction. □

This result also helps to compute the edges of the polygon efficiently:

Theorem 9. *The edges of a rotated orthogonally convex polygon can be reconstructed from its vertex coordinates in $O(n \log n)$ time.*

Proof. First, compute the convex hull $CH(S)$ of S in $O(n \log n)$ time. Then, for each edge e of S, test whether $H(e)$ is empty. This can be done by pre-computing the Voronoi diagram of S in $O(n \log n)$ time [6]. Here, the *Voronoi diagram* of n points S is a partition of the plane into n regions associated to the points, such that the points in each region are closer to the associated point than to any other point in S. For each edge, $H(e)$ is empty if and only if the nearest points of the midpoint of edge e are the endpoints of edge e and no other point of S. One query in the Voronoi diagram is necessary per edge, and each such query takes $O(\log n)$ time. Therefore, for all edges this can be read from the Voronoi diagram in $O(n \log n)$ total time.

By Lemma 7 there must exist an edge e with $H(e)$ empty; otherwise S is not the vertex set of a rotated orthogonally convex polygon. By Lemma 6, this edge e *must* be an edge of P in any solution P. So rotate S such that e becomes horizontal or vertical, and then apply O'Rourke's algorithm [5] to compute the edges of P. □

4.2. Point sets in 3D

In this section, we study Problem 2 in 3D, i.e., the following question: Given a set S of points in R^3, construct a rotated orthogonally convex polyhedron with vertex set S if one exists.

In order to reconstruct the edges and faces, a trivial algorithm is to compute the convex hull of the vertices of the polyhedron, and then try all possible rotations that make two non-parallel convex hull faces lie in orthogonal planes. For each rotation, one can apply the algorithm presented in Section 3 to reconstruct the edges and faces of the orthogonally convex polyhedron. This approach takes $O(n^3 \log n)$ time in the worst case, because there may be as many as $O(n^2)$ rotations that must be tried for $O(n)$ faces of the convex hull.

Unfortunately, it is not known whether there are multiple rotations that can allow an orthogonally convex polyhedron to be realized by the given vertex coordinates. Löffler and Mumford's result [16] almost seems to imply that this is the case, but their proofs require connectivity of the reconstructed graphs, while an orthogonally convex polyhedron need not have a connected graph.

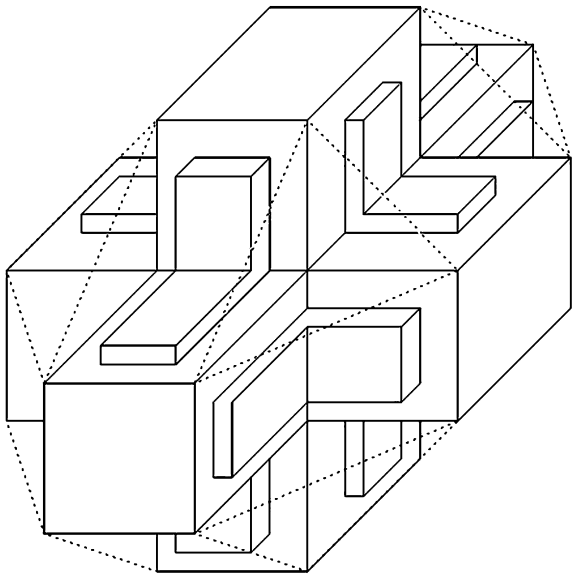

Fig. 5. An orthogonally convex polyhedron for which there exist no edges on the convex hull with empty balls.

But even without uniqueness, one can decrease the runtime and eliminate many of the rotations by applying the algorithm of Section 4.1 to each face of the convex hull. More precisely, assume that F is a face of the convex hull, and S_F is the set of all input points that lie in the plane of F. Apply the algorithm of Section 4.1 to S_F which takes $O(|S_F|\log|S_F|)$ time. If it succeeds, then the resulting rotated orthogonally convex polygon P_F can be a face of a rotated orthogonally convex polyhedron P, but only if the rotation is such that the face normal of F and edges of P_F become parallel to coordinate axes. After rotating S with this rotation, apply the algorithm of Section 3 to compute the edges and faces of the orthogonally convex polyhedron; this will reconstruct P if it exists. This takes $O(n\log n)$ time per face and $O(n^2\log n)$ overall for $O(n)$ faces.

Theorem 10. *The edges and faces of a rotated orthogonally convex polyhedron can be reconstructed from its vertex coordinates in* $O(n^2\log n)$ *time.*

There are some heuristics to decrease the number of rotations to be tried, but none of them leads to an improvement in the worst case. In particular, the three-dimensional equivalent of Lemma 6 holds for an edge e of $CH(S)$, if $H(e)$ is replaced by a ball $B(e)$ spanned by e. Thus if $B(e)$ is empty for some edge e (which can be tested in $O(n\log n)$ time), then there is only one possible rotation. Unfortunately, there are orthogonally convex polyhedra for which the 3D equivalent of Lemma 7 does *not* hold, so there need not always be an edge for which $B(e)$ is empty (see Fig. 5). Also, since Lemma 7 need not hold, it is not known whether the rotated orthogonally convex polyhedron in Theorem 10 is unique.

5. Reconstruction with additional points

In the paper thus far, we have studied an *exact* version of the problem, i.e., the given input set S must exactly be the set of vertices of the polygon/polyhedron to be reconstructed. In this section, we briefly consider the case when additional points are allowed. We do not have any ground-breaking results here, and mostly present a wealth of open problems.

Additional points could arise in a number of applications. In the object recognition application, it is possible that the method that created the data had some errors. In this case, additional points may be anywhere. In the data structures application, on the other hand, additional points would only be on the boundary or inside. Namely, it is common to describe a polygon/polyhedron via the polygonal curve/polyhedral surface that is its boundary, and many applications allow that the boundary is subdivided. This then creates additional points on the boundary. Furthermore, if the boundary of a polyhedron is subdivided, then the polygons on the boundary obtain additional points inside. We hence studied also the problem of additional points inside, both in 2D and 3D.

The problems to be studied hence are as follows: First, we need to clarify how the input set S is to be interpreted. Should it be exact, i.e., all points in S must be vertices of the computed polygon/polyhedron? Or may S have points in addition to the vertices? If the latter, must the additional points be on the boundary of the computed structure, or on the inside (including the boundary)?

Secondly, what do we know about the resulting polygon/polyhedron? Is it orthogonally convex, or a general orthogonal polygon/polyhedron? Is it in 2D (a polygon) or in 3D (a polyhedron)? Do we allow rotated polygons/polyhedra or not?

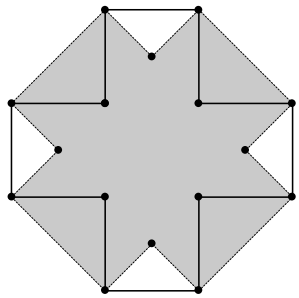

Fig. 6. Two rotated orthogonally convex polygons constructed from same input coordinates: additional points inside the polygons are allowed.

This raises 24 problems. For each of them, we want to know whether it is possible to construct an orthogonal polygon/polyhedron in polynomial time, and if so, what the fastest time complexity is. We are also interested in whether the answer is unique.

For these many problems, we will only provide a selection of answers below:

- Rappaport [4] showed that it is NP-hard to reconstruct an orthogonal polygon from a set of 2D points if additional points on the boundary are allowed. This NP-hardness transfers to rotated polygons, since there are only $O(n)$ possible rotations.
- Orthogonally convex polygons can be reconstructed even with additional points on the boundary and/or inside by applying the orthogonally convex hull algorithm, which takes $O(n \log n)$ time [13] and the answer is unique.
- For rotated orthogonally convex polygons, the above can be applied to all $O(n)$ possible rotations, so in $O(n^2 \log n)$ time we can find an answer for orthogonally convex polygons with additional points inside. The answer is not unique, see Fig. 6.
- For rotated orthogonally convex polygons, if additional points are only on the boundary, then the answer is unique and can be found in $O(n \log n)$ time.
  Namely, the algorithm for Theorem 9 (reconstruct a rotated orthogonally convex polygon from the exact set of vertices) works even if there are additional points on the boundary of the polygon. The only small modification needed is that after the computation of the convex hull, one must merge all collinear convex hull edges. Then the correct rotation can be computed as discussed in Section 4.1, and the answer can be found by computing the orthogonally convex hull after rotating the point set.
- For orthogonally convex polyhedra, the algorithm presented in Section 3 for reconstruction from the exact set of vertices works even if there are additional points inside or on the boundary of the polyhedron. This is because the additional points do not affect the shadows created at each layer, except that each such point may be in a separate layer. The overall runtime is still $O(n \log n)$: each additional point contributes $O(\log n)$ to shadow updates. Doing so the shadows are not changed and the point is identified as an additional point.
- For rotated orthogonally convex polyhedra, the above can be applied to all possible $O(n^2)$ rotations and yields an answer in $O(n^3 \log n)$ time. The answer is again not unique, which can be seen by extruding the example of Fig. 6.

Table 1 summarizes the results in this paper.

6. Conclusion and further remarks

In this paper, we studied the problem of reconstructing a polygon or polyhedron given only its set of vertices. We provided efficient algorithms for orthogonally convex polygons/polyhedra, both if points must be used as they are, and if points are allowed to be rotated.

We briefly studied the case of additional points, which leaves many open problems already. Other interesting open problems are as follows:

- Given a set S of points in 3D, is there a unique rotated orthogonally convex polyhedron with S as its vertices? This holds if the graph of the polyhedron is required to be connected [16], but what is the situation in general?
- What are algorithms to reconstruct orthogonal polyhedra that are not orthogonally convex? Often the answer here will not be unique, but how difficult is it to construct at least one? Can the NP-hardness proof by Rappaport [4] for 2D-reconstruction with additional points be used to prove NP-hardness of 3D-reconstruction with the exact point set?

Finally, the running time remains to be improved for many of the problems studied.

Table 1
Summary of results. The first row in each cell shows the runtime of the fastest known algorithm that computes the edges (and faces), if an algorithm exists. The second row is either a citation of an existing work that presented this algorithm, or a reference to a section in this paper. The third row shows whether the computed edges (and faces) using this algorithm are unique or not.

	Non-rotated		Rotated	
	2D	3D	2D	3D
Orthogonal				
Vertices of	$O(n \log n)$	?	$O(n^2)$	?
polygon/	[5]		Section 4.1	
polyhedron	unique	not unique	unique [16]	not unique
Additional	NP-hard	?	NP-hard	?
points on	[4]		[4]	
boundary	not unique	not unique	not unique	not unique
Additional	?	?	?	?
points				
inside	not unique	not unique	not unique	not unique
Orthogonally convex				
Vertices of	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$
polygon/	[5]	Section 3	Section 4.1	Section 4.2
polyhedron	unique	unique	unique	?
Additional	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n^3 \log n)$
points on	[13]	Section 5	Section 5	Section 5
boundary	unique	unique	unique	not unique
Additional	$O(n \log n)$	$O(n \log n)$	$O(n^2 \log n)$	$O(n^3 \log n)$
points	[13]	Section 5	Section 5	Section 5
inside	unique	unique	unique	not unique

References

[1] J. O'Callaghan, Computing the perceptual boundaries of dot patterns, Comput. Graph. Image Process. 3 (2) (1974) 141–162.
[2] N. Ahuja, Dot pattern processing using Voronoi neighborhoods, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-4 (3) (1982) 336–343.
[3] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C.T. Silva, Point set surfaces, in: Proceedings of the Conference on Visualization '01, VIS '01, IEEE Computer Society, Washington, DC, USA, 2001, pp. 21–28.
[4] D. Rappaport, On the complexity of computing orthogonal polygons from a set of points, Tech. Rep. TR-SOCS-86.9, McGill University, 1986.
[5] J. O'Rourke, Uniqueness of orthogonal connect-the-dots, in: Computational Morphology, Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1988, pp. 97–104.
[6] F. Preparata, M. Shamos, Computational Geometry, An Introduction, Springer-Verlag, New York, 1985.
[7] O. Bournez, O. Maler, A. Pnueli, Orthogonal polyhedra: Representation and computation, in: F. Vaandrager, J. van Schuppen (Eds.), Hybrid Systems: Computation and Control, in: Lecture Notes in Computer Science, vol. 1569, Springer, Berlin, Heidelberg, 1999, pp. 46–60.
[8] H. Steinhaus, One Hundred Problems in Elementary Mathematics, Dover Publications, New York, 1964.
[9] P.K. Agarwal, F. Hurtado, G.T. Toussaint, J. Trias, On polyhedra induced by point sets in space, Discrete Appl. Math. 156 (2008) 42–54.
[10] B. Grünbaum, Hamiltonian polygons and polyhedra, Geombinatorics 3 (1994) 83–89.
[11] A. Aguilera, D. Ayala, Orthogonal polyhedra as geometric bounds in constructive solid geometry, in: Proceedings of the Fourth ACM Symposium on Solid Modeling and Applications, SMA '97, ACM, New York, NY, USA, 1997, pp. 56–67.
[12] A. Aguilera, D. Ayala, Orthogonal polyhedra: Study and application, PhD thesis, Universitat Politecnica de Catalunya, 1998.
[13] T. Ottmann, E. Soisalon-Soininen, D. Wood, On the definition and computation of rectilinear convex hulls, Inform. Sci. 33 (3) (1984) 157–171.
[14] G.J.E. Rawlins, D. Wood, Ortho-convexity and its generalizations, in: Computational Morphology, Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1988, pp. 137–152.
[15] D.G. Kirkpatrick, R. Seidel, Output-size sensitive algorithms for finding maximal vectors, in: Proceedings of the First Annual Symposium on Computational Geometry, SCG '85, ACM, New York, NY, USA, 1985, pp. 89–96.
[16] M. Löffler, E. Mumford, Connected rectilinear graphs on point sets, in: I. Tollis, M. Patrignani (Eds.), Graph Drawing, in: Lecture Notes in Computer Science, vol. 5417, Springer, Berlin, Heidelberg, 2009, pp. 313–318.