Reconstructing Orthogonal Polyhedra from Putative Vertex Sets

Therese Biedl[1] and Burkay Genc[1]

David R. Cheriton School of Computer Science,
University of Waterloo, Waterloo ON N2L 3G1, Canada
biedl@uwaterloo.ca, bgenc@uwaterloo.ca

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. We are especially interested in testing whether such a polyhedron is unique. Since this is not the case in general, we focus on orthogonally convex polyhedra, and give an $O(n \log n)$ algorithm to find the answer. We then consider the case where the given set of points may be rotated beforehand. For 2D, we prove uniqueness and provide an $O(n \log n)$ algorithm to obtain the answer, which can then be used for an $O(n^2 \log n)$ algorithm in 3D.

Key words: vertex representation, orthogonal polyhedra, 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 of orthogonal polyhedra and their vertices will be given in Section 2.)

Motivation and our results Our interest in the problems arises from the question of possible representations of orthogonal polyhedra. The "standard" representation of polyhedra (representing the graph as doubly-connected edge list (see [12]) and vertex coordinates) contains much redundancy when applied to an orthogonal polyhedron. What information can be omitted while maintaining uniqueness of the polyhedron?

We are specifically interested in the *vertex representation*, where we store the vertex coordinates and nothing else. Our problem can hence be re-phrased as follows: Is the vertex representation unambiguous, i.e., does it give rise to one unique orthogonal polyhedron?

Bournez et al. [3] gave an example where this representation is ambiguous, even in 2D (see Figure 1). In this example, the polygon boundary touches itself repeatedly, which is not usually allowed. We show that if we forbid this, then

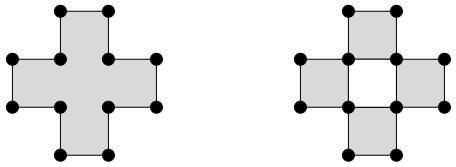

Fig. 1. Two polygons (one touching itself) that have the same set of vertices. From [3].

there is a simple, $O(n \log n)$, algorithm, to test whether a set of 2D points belongs to an orthogonal polygon.

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

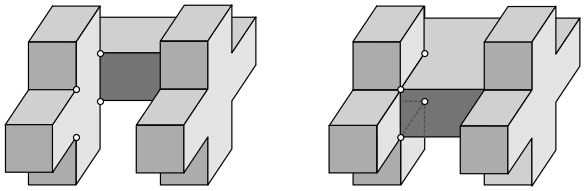

Fig. 2. Two orthogonal polyhedra that have the same vertex set.

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.

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. We also show that if such a polygon exists, then it is unique. Then we use this to reduce the time complexity for 3D to $O(n^2 \log n)$.

Related results Reconstructing polygons from a given point set was first proposed by Steinhaus [13]. Unless the points are collinear, there always exists a polygon, and it can be found in many ways; see [8] for 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 [7]; see [8] for 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 [1, 2] 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 (the extreme vertices must "pair up", similarly as in our algorithm for 2D polygons in Section 3.1). Bournez, Maler and Pnueli [3] 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. (Their technique works for arbitrarily high dimensions, and the runtime then becomes $O(nd^2(\log n + 2^d))$.) 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 *polygonal curve* is a simple closed curve that consists of a finite number of line segments. A *polygon* is a set P in a plane whose boundary ∂P is a polygonal curve. 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 $S \subseteq R^d$ is *convex* if $x \in S$ and $y \in S$ implies that the segment $[xy]$ is entirely in S. For a set $S \subseteq R^d$, we define the convex hull $CH(S)$ to be the intersection of all convex sets that contain S. $CH(S)$ is a convex polygon for any bounded set 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 for any two points in P that determine a horizontal or vertical line segment, this line segment is entirely in P.

Defining polyhedra precisely is nontrivial. Let a *closed polyhedral surface* be the union of a finite set of polygons that satisfies the following properties [4, 5]: (1) Any pair of polygons meet only at their edges or vertices. (2) Each side of each polygon meets exactly one other polygon along an edge. (3) It is possible to travel from the interior of any polygon to the interior of any other. (4) Let v be any vertex and let $F_1, F_2, \ldots, F_k$ be the k polygons which meet at v. It is possible to travel over the polygons F_i from one to any other without passing through v. See the books by Coxeter, Cromwell or O'Rourke [4, 5, 10] for detailed discussions of these properties. The vertices and edges of the closed polyhedral surface are the vertices and edges of the polygons that define it. A *polyhedron* is then a subset of R^3 whose boundary is a closed polyhedral surface.

To obtain the *faces* of the polyhedron, we merge co-planar adjacent polygons of the surface. A face may or may not be a polygon (because its boundary may touch itself, or its boundary may consist of multiple polygonal chains), but we

define its vertices and edges similar to polygons: A vertex of a face is a place where its boundary changes direction, or where the boundary touches itself, or a point where some other face has a vertex. An edge of a face is a line segment on its boundary that connects two of its vertices. The doubly-connected edge list stores the list of edges and their incidences to vertices and faces.

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. An *orthogonally convex polyhedron* is a polyhedron whose intersection with every orthogonal plane is either empty or a single orthogonally convex polygon.

An orthogonally convex hull of a point set may be defined in different ways which are not all equivalent (see [11] for details for 2D). We use the following definition. An *orthogonal line* is a line that is parallel to a coordinate axis. An *orthogonal half-space* is a a set S such that for every orthogonal line ℓ, the intersection of ℓ with S is a line, a ray, or empty. (Fink and Wood [6] study properties of these half-spaces in detail.) The *orthogonally convex hull* of a set S is then the intersection of all orthogonal half-spaces that contain S.

3 Reconstructing without rotation

In this section, we first study the problem of reconstructing orthogonal polygons in 2D, given a set of points. This has been implicitly solved by Aguilera and Ayala [2], and we mostly review it here. Then we give an $O(n \log n)$ algorithm for the 3D problem, and also explain why other approaches do not work.

3.1 Point sets in 2D

So, assume we are given a set S of points, and we want to find an orthogonal polygon for which this is the set of vertices. Let $s_1, \ldots, s_t$ be a maximal set of points in S that have the same y-coordinate, sorted by increasing x-coordinate. Since every vertex is incident to exactly one horizontal edge, (s_i, s_{i+1}) must be an edge for odd values of i and is not an edge for the rest (see Figure 3).

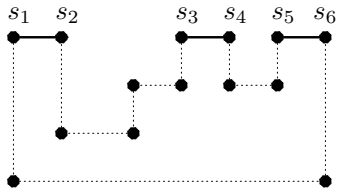

Fig. 3. Reconstruction in 2D: There must be an edge between odd-indexed and even-indexed vertices in each row and column.

Repeating this for every set of points with the same x-coordinates, and in a symmetric fashion for every set of points with the same y-coordinates, we obtain

exactly which pairs of vertices must form edges and which pairs are not allowed to be edges. The resulting set of edges either surrounds a polygon (then we are done) or it doesn't (then no such polygon can exist.) Clearly the algorithm can be implemented in $O(n \log n)$ time by sorting the points suitably twice and then scanning them in order.

The only property that we have used is that each vertex has exactly one vertical and one horizontal incident edge. Therefore, even the existence of multiple orthogonal polygons with disjoint boundaries may be tested using our algorithm.

Theorem 1. *For a set of n points in 2D, we can determine in $O(n \log n)$ time whether there exists a set of polygons with disjoint boundaries whose vertics are exactly S. Such a set is unique.*

3.2 Points sets in 3D

Now we study Problem 1 in 3D, and give an $O(n \log n)$ time algorithm that uses $O(n)$ space.

Approaches that do not work We first want to mention two simple approaches, of which one does not work at all, and the other works only partially.

The first idea would be to emulate the algorithm for 2D as follows: Compute for each orthogonal plane the set of points that is in it. Whenever there are points, use the 2D algorithm to identify the edges within that plane, which then defines the vertices of the polyhedron. Unfortunately, this is not correct. In the 2D algorithm, the crucial step was that every vertex has exactly one horizontal and one vertical edge. The equivalent statement is not true in 3D: A vertex may have one or two incident edges parallel to a coordinate axis. Even worse, it is actually not possible to identify all faces within one orthogonal plane by only looking at the vertices in that plane; see Figure 4.

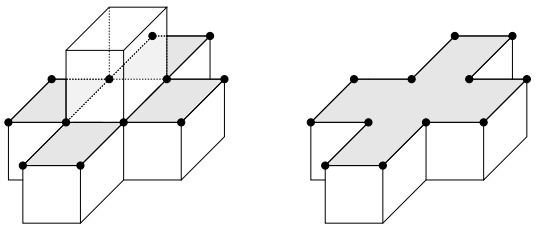

Fig. 4. Two polyhedra may have the same set of vertices, but different faces, within one orthogonal plane.

The second idea would be to compute the orthogonally convex hull of the given point set S, and to test whether its vertex set is exactly S. This approach is principally correct: One can show that if there is an orthogonally convex

polyhedron P with vertex set S, then indeed the orthogonally convex hull of S is P. However, there appears to be very little work done on efficiently computing the orthogonally convex hull. The question "given a set S, are all these points on the convex hull" can be answered in $O(n \log n)$ time by computing whether each point is a local maximum in one of the 8 directions. But there appear to be no results in the literature that truly reconstruct the orthogonally convex polyhedron, including all vertex and edge information, in $O(n \log n)$ time.

Our approach Our algorithm consists of two steps. In the first step, we sweep through all six orthogonal directions and construct the *shadows* of parts of the polyhedron. In the second step, we extract the actual edges and faces of the polyhedron from these shadows. Once the actual edges and faces are reconstructed, we can describe a closed polyhedral surface and thus the polyhedron is reconstructed.

To explain this in more detail, we need some definition. Assume that we have an orthogonally convex polyhedron P. Let $x_1 < x_2 < \ldots < x_t$ be the values for which some point in S has x-coordinate x_i. For $i = 1, \ldots, t$, let P_i^- be the left half of the polyhedron obtained when slicing P with a plane $\{x = x_i + \varepsilon\}$ (for a small $\varepsilon > 0$), and let π_i be the projection of P_i^- onto an x-plane; we call π_i the *i^{th} shadow in x^+-direction*. See Figure 5. We define π_0 to be the empty set. The following lemmas, whose fairly straightforward proofs we omit for space reasons, are crucial to our algorithm:

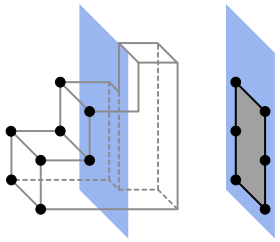

Fig. 5. The definition of π_2.

Lemma 1. *Let P be an orthogonally convex polyhedron. Then each π_i, $i = 1, \ldots, t$ is an orthogonally convex polygon. Moreover, π_i is the orthogonally convex hull of all vertices in P_i^- projected onto an x-plane.*

Lemma 2. *Let P be an orthogonally convex polyhedron. For $i = 1, \ldots, t$, let F_i be the set of all faces whose normal is the negative x-axis and whose x-coordinate is x_i. Then the union of the faces in F_i is exactly the closure of $\pi_i - \pi_{i-1}$.*

Our algorithm can now be outlined as follows. Assume that we have sorted the points by x-coordinates. Let $x_1 < \ldots < x_t$ be the x-coordinates for which

a vertex with this x-coordinate exists, and let the *ith layer* be all vertices with x-coordinate x_i. For $i = 1, \ldots, t$, compute the (2D) orthogonally convex hull of the projections of the first i layers onto an x-plane. We call the result π_i, since by Lemma 1 this is exactly the ith shadow if there exists a polyhedron P. Next, compute $\pi_i - \pi_{i-1}$; this then computes the faces as in Lemma 2 if P exists. Repeating this for all 6 directions, we obtain all faces (and can obtain edge information and incidence structure from there.) This computes the only possible set of faces that could belong to an orthogonally convex polyhedron P with vertex set S. Now we need to verify that these faces indeed form a polyhedron (i.e., check the properties of a polyhedral surface), and that the vertices that they define is exactly set S.

This algorithm can easily be implemented in $O(n^2 \log n)$ time if we use the (2D) orthogonally convex hull algorithm by Ottman et al. [11], 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 computing the faces. Given a set of points with its orthogonally convex hull, it is easy to update the orthogonally convex hull in $O(\log n)$ time per added point. (This is well-known for general convex hulls, see e.g. [12], and can easily be modified to handle orthogonally convex hulls.)

To compute $\pi_i - \pi_{i-1}$, observe that all connected components of $\pi_i - \pi_{i-1}$ are incident to at least one point in layer i. Hence, exploring the boundary of π_i from vertices in layer i, we can walk around each connected piece of $\pi_i - \pi_{i-1}$, hence find each face F 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. The final verification steps can also be implemented in $O(n)$ time. Once we are finished with the ith layer, we can discard π_{i-1} which is no longer needed, hence the space complexity is $O(n)$ as well.

Theorem 2. *For a set of n points in 3D, we can determine in $O(n \log n)$ time whether there exists an orthogonally convex polyhedon whose vertex set is exactly S. Such a polyhedron is unique.*

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 polygon/polyhedron. The following notation will be helpful: An *α-orthogonal polygon* (for $0 \leq \alpha < \pi/2$) 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 $0 \leq \alpha < \pi/2$. 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.*

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, and for each of them, running the algorithm of Section 3.1. To find the possible angles, compute the convex hull $CH(S)$ of S. For each of the edges on the convex hull, try the angle α that makes this edge horizontal or vertical. Clearly, if P exists, then one of these rotations must find it because of the following observation:

Lemma 3. *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, and they are edges of $CH(P)$.*

The time complexity of this approach is $O(n^2 \log n)$ for an input set of n points, since the convex hull has at most n edges.

Theorem 3. *For a set S of n points in 2D, we can determine in $O(n^2 \log n)$ time whether there exists a rotated orthogonal polygon whose set of vertices is exactly S.*

For orthogonally convex polygons, with some pre-computation we can eliminate all but one possible rotation, and hence improve the time complexity to $O(n \log n)$. To do so, we first show that in fact only one rotation is possible.

Given an edge e of a convex polygon C, we define $H(e)$ to be the closed half-circle of e that intersects C. See also Figure 6(a).

Lemma 4. *Let P be an α-orthogonally convex polygon for some $0 \leq \alpha < \pi/2$, and let $e = (v, w)$ be an edge of $CH(P)$. If e is not an edge of P, then $H(e)$ contains at least one vertex $\neq v, w$ of P.*

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 Figure 6(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 is a polygonal curve, which is simple and must not cross itself, a contradiction. □

We need a second 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. Lemma 3.) We call these the *extreme edges* of P. In what follows, the indices of $e_0, \ldots, e_3$ will be taken modulo 4, and $||e_i||$ denotes the length of e_i.

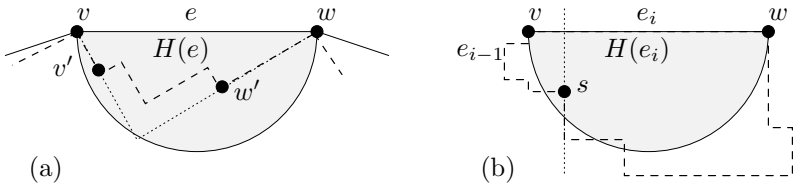

Fig. 6. (a) $H(e)$ must contain another vertex. In this picture, $\alpha \approx \pi/6$. Polygon P is dashed. (b) e_{i-1} can at most be half as long as e_i. Polygon P is dashed.

Lemma 5. *Let P be a rotated orthogonally convex polygon with extreme edges $e_0, \ldots, e_3$. For any i, if $H(e_i)$ contains some vertex of P other than the endpoints of e_i, then $||e_i|| \geq 2\min\{||e_{i-1}||, ||e_{i+1}||\}$.*

Proof. After possible rotation, e_i is horizontal and the rest of P has smaller y-coordinates than e_i. Let v and w be the endpoints of e_i with v left of w. Let $s \neq v, w$ be a vertex of P inside $H(e_i)$. See also Figure 6(b). The intuition behind the lemma is that s is on the "other side" of P, which imposes restrictions on how long the neighbouring extreme edges can be.

To prove this formally, consider the vertical line through s. This line intersects P at s and somewhere along edge (v, w). It cannot intersect 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. Assume that the monotone chain is the one that contains s and v but not w; this chain contains e_{i-1}. (The other case is similar, yielding a bound on e_{i+1}.) Since the chain is monotone, edge e_{i-1} (which is vertical) cannot be longer than the distance from s to (v, w), which is at most $||e_i||/2$ since s is inside $H(e_i)$. □

The contrapositive of this lemma implies that if e_i is the shortest extreme edge, then $H(e_i)$ must not contain any other vertex of the polygon, which yields the following crucial corollary:

Corollary 1. *Let P be a rotated orthogonally convex polygon. Then for at least one extreme edge e of P, $H(e)$ contains no other vertex of P.*

This corollary and Lemma 4 now imply uniqueness.

Theorem 4. *For a set S of points in 2D, there can exist at most one rotated orthogonally convex polygon whose set of vertices is exactly S.*

Proof. Assume for contradiction that S is the vertex set of both an α-orthogonally convex polygon P and an α'-orthogonally convex polygon P' where $0 \leq \alpha \neq \alpha' < \pi/2$. By Corollary 1, 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 4, $H(e)$ contains some vertex of P', which is a point of S, a contradiction. □

The same results also help to find such a polygon efficiently, if one exists.

Theorem 5. *For a set S of n points in 2D, we can determine in $O(n \log n)$ time whether there exists a rotated orthogonally convex polygon whose set of vertices is exactly S.*

Proof. First, compute the convex hull $CH(S)$ of S; this takes $O(n \log n)$ time (see e.g. [12].) 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 (see e.g. [12].) 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. This can be read from the Voronoi-diagram in $O(n \log n)$ total time.

Now we can read the appropriate rotation from whether the half-circles are empty. Namely, if $H(e)$ is empty for some edge e, then by Lemma 4 e *must* be an edge of any rotated orthogonally convex polygon P on this point set. So rotate S such that e becomes horizontal or vertical, and then apply the algorithm from Section 3.1 to determine whether there is indeed such a polygon. This is the only rotation that could possibly work, so if any $H(e)$ is empty, we can determine whether a polygon exists in $O(n \log n)$ time. If none of $H(e)$'s is empty, then by Corollary 1, no polygon P can exist and we are done. □

4.2 Points in 3D

For point sets in 3D, the trivial algorithm (try all possible rotations) takes $O(n^3 \log n)$, and even then only works for orthogonally convex polyhedra. Namely, in order to fix a rotation, we need to fix two face normals. Hence there are $O(n^2)$ pairs of normals to choose (among the $O(n)$ face normals of the convex hull of the input set). For each of them, applying our $O(n \log n)$ algorithm from Section 3.2 after a suitable rotation yields an $O(n^3 \log n)$ algorithm to reconstruct a rotated orthogonally convex polyhedron if one exists.

We can eliminate many of these rotations by applying the 2D algorithm 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 F. Apply the 2D algorithm (for rotated point sets) to S_F which takes $O(|S_F| \log |S_F|)$ time. If it succeeds, then the resulting rotated orthogonally convex polygon P_F could be an extreme face of a solution polyhedron P, but only if the rotation is such that the edges of P_F and the face normal of F become coordinate axes. After rotating S with this rotation, apply the algorithm for 3D non-rotated point sets; this will reconstruct P if it exists. We need to repeat this for every face of the convex hull where the 2D algorithm succeeds, so the time complexity is $O(n^2 \log n)$.

Theorem 6. *For a set S of n points in 3D, we can determine in $O(n^2 \log n)$ time whether there exists a rotated orthogonally convex polyhedron whose set of vertices is exactly S.*

We know of some heuristics to decrease the number of rotations that must be tried, but none of them leads to an improvement in the worst case. In particular, the 3D equivalent of Lemma 4 holds if we replace $H(e)$ by a ball $B(e)$ spanned

by e for every edge e on $CH(S)$. 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 orthogonaally convex polyhedra for which the 3D equivalent of Corollary 1 does *not* hold, so there need not always be an edge for which $B(e)$ is empty. Also, since Corollary 1 need not hold, we do not know whether the rotted orthogonally convex polyhedron in Theorem 6 is unique.

5 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 we must use points as they are, and if we are allowed to rotate points. For 2D, we also showed that there can be only one rotation that works.

In our problem, we did not allow any points other than the vertices of the polyhedron. If our input results from taking the vertices of a polyhedral surface that described a polyhedron, then we may have extra vertices that are on the surface of the polyhedron. Hence we are also interested in the following problem:

Problem 3. Given a set S of points in 3D, is there an orthogonal polyhedron P for which every vertex is in S, and all points in S are on the surface of P?

All our results for orthogonally convex polygons and polyhedra hold for this problem as well. The 2D algorithm in Section 3.1 does not work (more below), but to reconstruct an orthogonally convex polygon from a set of points, we simply compute the orthogonally convex hull in $O(n \log n)$ time; see [11]. Our 3D algorithm from Section 3.2 works without changes, since the additional points will not change the computed shadows at all. Our 2D algorithm from Section 4.1 works as long as we remove all points from S that are on $CH(S)$, but not vertices of $CH(S)$. Therefore, the 3D algorithm from Section 4.2 works as well.

For the non-orthogonally convex case, Problem 3 is difficult. The answer need not be unique, even in 2D, see Figure 7. With a fairly simple reduction from Hamiltonian Cycles in Grid Graphs (which is NP-hard [9]), one can in fact show that Problem 3 becomes NP-hard, even in 2D, for orthogonal polygons.

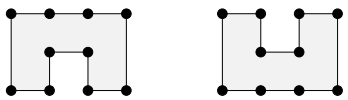

Fig. 7. The 2D case may have multiple answers when additional vertices are allowed.

The main remaining open problem concerns non-orthogonally convex polygons and polyhedra when we are given the exact vertex set. Can we improve the running time in Theorem 3 to $O(n \log n)$ as well? Is the polygon unique in this case? Our proof does not carry over easily: Lemma 4 extends to non-orthogonally

convex polygons (with some small modifications), but there are orthogonal polygons (not orthogonally convex) for which Corollary 1 does not hold. So is there a 2D point set that is the vertex set of two different rotated orthogonal polygons?

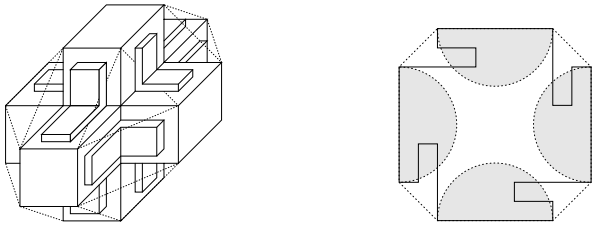

Fig. 8. An orthogonally convex polyhedron P and a non-orthogonally convex polygon for which the equivalent of Corollary 1 does not hold.

Finally, in 3D the vertex representation is ambiguous for non-orthogonally covex polyhedra. But given a point set, can we reconstruct *some* orthogonal polyhedron for which this is the set of vertices? Or is this NP-hard?

References

1. A. Aguilera and D. Ayala. Orthogonal polyhedra as geometric bounds in constructive solid geometry. In *Proceedings of the Fourth ACM Symposium on Solid Modeling and Applications*, pages 56–67. ACM Press, 1997.
2. A. Aguilera and D. Ayala. *Orthogonal Polyhedra: Study and Application*. PhD thesis, Universitat Politecnica de Catalunya, 1998.
3. O. Bournez, O. Maler, and A. Pnueli. Orthogonal polyhedra: Representation and computation. In *Hybrid Systems: Computation and Control*, volume 1569, pages 46–60, Nijmegen, The Netherlands, 29–31 March 1999.
4. H.S.M. Coxeter. *Regular polytopes*. Dover Publications, New York, 1973.
5. P.R. Cromwell. *Polyhedra*. Cambridge University Press, June 1997.
6. Eugene Fink and Derick Wood. Generalized halfspaces in restricted-orientation convexity. *Journal of Geometry*, 62:99–120, 1998.
7. B. Grünbaum. Hamiltonian polygons and polyhedra. *Geombinatorics*, 3:83–89, January 1994.
8. F. Hurtado, G.T. Toussaint, and J. Trias. On polyhedra induced by point sets in space. In *Canadian Conference on Computational Geometry*, pages 107–110, 2003.
9. A. Itai, C.H. Papadimitiou, and J.L. Szwarcfiter. Hamiltonian paths in grid graphs. *SIAM Journal of Computing*, 11(4):676–686, 1982.
10. J. O'Rourke. *Computational geometry in C (2nd ed.)*. Cambridge University Press, New York, NY, USA, 1998.
11. T. Ottmann, E. Soisalon-Soininen, and D. Wood. On the definition and computation of rectilinear convex hulls. *Information Sciences*, 33:157–171, 1984.
12. Franco P. Preparata and Michael I. Shamos. *Computational geometry: an introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1985.
13. Hugo Steinhaus. *One Hundred Problems in Elementary Mathematics*. Dover Publications, New York, 1964.