Epsilon-Unfolding Orthogonal Polyhedra

Mirela Damian[1], Robin Flatland[2], Joseph O'Rourke[3]

[1] Dept. Comput. Sci., Villanova University, e-mail: mirela.damian@villanova.edu
[2] Dept. Comput. Sci., Siena College, e-mail: flatland@siena.edu
[3] Dept. Comput. Sci., Smith College, e-mail: orourke@cs.smith.edu [*]

Abstract. An *unfolding* of a polyhedron is produced by cutting the surface and flattening to a single, connected, planar piece without overlap (except possibly at boundary points). It is a long unsolved problem to determine whether every polyhedron may be unfolded. Here we prove, via an algorithm, that every *orthogonal polyhedron* (one whose faces meet at right angles) of genus zero may be unfolded. Our cuts are not necessarily along edges of the polyhedron, but they are always parallel to polyhedron edges. For a polyhedron of n vertices, portions of the unfolding will be rectangular strips which, in the worst case, may need to be as thin as $\varepsilon = 1/2^{\Omega(n)}$.

Key words. general unfolding, grid unfolding, orthogonal polyhedra, genus-zero

1. Introduction

Two unfolding problems have remained unsolved for many years [4]: (1) Can every convex polyhedron be edge-unfolded? (2) Can every polyhedron be unfolded? An *unfolding* of a 3D object is an isometric mapping of its surface to a single, connected planar piece, the "net" for the object, that avoids overlap. An *edge-unfolding* achieves the unfolding by cutting edges of a polyhedron, whereas a *general unfolding* places no restriction on the cuts. General unfoldings are known for convex polyhedra, but not for nonconvex polyhedra. It is known that some nonconvex polyhedra cannot be edge-unfolded, but no example is known of a nonconvex polyhedron that cannot be unfolded with unrestricted cuts. The main result of this paper is that the class of genus-zero orthogonal polyhedra has a general unfolding. As we only concern ourselves with general unfoldings of genus-zero polyhedra in this paper, we will drop the "general" and "genus-zero" modifiers when clear from the context.

[*] Supported by NSF Distinguished Teaching Scholars award DUE-0123154.

The difficulty of the unfolding problem has led to a focus on *orthogonal polyhedra*—those whose faces meet at angles that are multiples of 90°—and especially on genus-zero polyhedra, i.e., those whose surfaces are homeomorphic to a sphere. This line of investigation was initiated in [1], which established that certain subclasses of orthogonal polyhedra have an unfolding: *orthostacks* and *orthotubes*. Orthostacks are extruded orthogonal polygons stacked along one coordinate direction. The orthostack algorithm does not achieve an edge unfolding, but it is close, in a sense we now describe.

A *grid unfolding* adds edges to the surface by intersecting the polyhedron with planes parallel to Cartesian coordinate planes through every vertex. This concept has been used to achieve grid *vertex unfoldings* of orthostacks [3], and later, grid vertex unfoldings of all genus-zero orthogonal polyhedra [8]. (A "vertex unfolding" is a loosening of the notion of unfolding that we do not pause to define [2].) A $k_1 \times k_2$ *refinement* of a surface [5] partitions each face further into a $k_1 \times k_2$ grid of faces; thus a 1×1 refinement is an unrefined grid unfolding. The orthostack algorithm achieves a 2×1 refined grid unfolding. It remains open to achieve a grid unfolding of orthostacks. (It is known that not all orthostacks may be edge unfolded.)

The algorithm we present in this paper could be characterized as achieving a $2^{O(n)} \times 2^{O(n)}$ refined grid unfolding of orthogonal polyhedra of n vertices. We coin the term *epsilon-unfolding* to indicate a refinement with no constant upper bound, but which instead grows with n. In our case, some portions of the unfolding might be ε-thin, with $\varepsilon = 1/2^{\Omega(n)}$.

Our algorithm has it roots in the staircase unfolding of [1], in the spiral strips used in [6], and the band structure exploited in [8], but introduces several new ideas, most notably a recursive spiraling pattern whose nesting leads to the ε-thin characteristic of the unfolding.

1.1. Definitions

Let O be a solid, genus-zero, orthogonal polyhedron, with edges parallel to the x, y and z axes of a Cartesian coordinate system. We use the following notation to describe the six types of faces of O, depending on the direction in which the outward normal points: *front*: $-y$; *back*: $+y$; *left*: $-x$; *right*: $+x$; *bottom*: $-z$; *top*: $+z$. We take the z-axis to define the vertical direction. The spiral paths that play a key role in our algorithm will "move" in the y-direction, wrapping around {top, right, bottom, left} faces.

Let Y_i be the plane $y = y_i$ orthogonal to the y-axis. Let $Y_0, Y_1, \dots, Y_i, \dots$ be a finite sequence of parallel planes passing through every vertex of O, with $y_0 < y_1 < \cdots < y_i < \cdots$. We call the portion of O between planes Y_i and Y_{i+1} *layer i*; it includes a collection of disjoint connected components of O. Referring to Figure 1, layers 0 and 2 each contains one component (D and A, respectively), whereas layer 1 contains two components (B and C). The surface pieces that surround a component are called *bands* (labeled in Figure 1). Each band has two *rims*, the cycle of edges that lie in its bounding Y_i and Y_{i+1} planes. Each

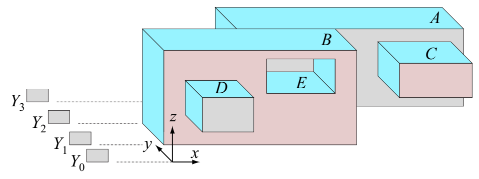

Fig. 1. Definitions: A, B, C, and D are protrusions; E is a dent.

component is bounded by an outer band, but it may also contain inner bands, bounding holes. Outer bands are called *protrusions* and inner bands are called *dents* (E in Figure 1).

1.2. Overview

The algorithm first partitions the polyhedron O into layers with the Y_i planes. It then forms an "unfolding tree" T_U whose nodes are bands, and with a parent-child arc representing a "z-beam" of visibility lying in a xz-face that connects the bands in their shared Y_i plane. Front and back children are distinguished according to the relative y-positions of the children with respect to the parent. The recursion follows a preorder traversal of this tree. A thin spiral path winds around the {top, right, bottom, left} faces of a root band b, visits each of the front children recursively, and then each of the back children recursively. The children are visited in a parentheses-nesting order that is forced by the turn-around requirements. The spiral alternates turns so that its unfolding to the plane is a staircase-like path monotone with respect to the horizontal (cf. Figure 3). When the path finishes spiraling around the last back child of b, it is deeply nested inside the spiral, and must retrace the entire path to return adjacent to its starting point. (It is this retracing, recursively encountered, that causes the exponential thinness.) Again this is accomplished while maintaining the staircase-like layout. Finally, the front and back faces are hung above and below the staircase. Following the physical model of cutting out the net from a sheet of paper, we permit cuts representing *edge overlap*, where the boundary touches but no interior points overlap. This can occur when hanging front and back faces off the spiral strip.

2. Unfolding Extrusions

It turns out that nearly all algorithmic issues are present in unfolding polyhedra that are extrusions of simple orthogonal polygons. Therefore we will describe the algorithm for this simple shape class first, in detail, and then show that the ideas extend directly to unfolding all orthogonal polyhedra.

Let O be a polyhedron that is an extrusion in the z direction of a simple orthogonal polygon. We start with the partition π of O induced by the Y_i planes passing through every vertex, as described in Section 1. Each element in the partition is a box surrounded by a four-face band. The dual graph of π has a node for each band and an edge between each pair of adjacent bands. For ease of presentation, we will use the terms *node* and *band* interchangeably. Because O is simply connected, the dual graph is a tree T_U, which we refer to as the *unfolding tree*. The root of T_U is any band that intersects Y_0. See Figure 2.

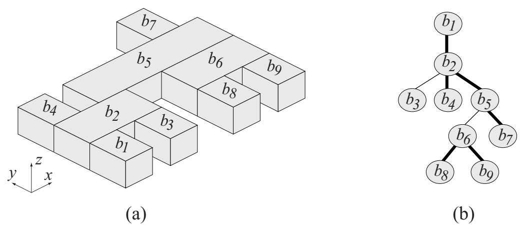

Fig. 2. (a) O partitioned by Y_i planes; the bands are labeled. (b) Unfolding tree T_U.

We distinguish between the two rims of each band via a recursive classification scheme. The rim of the root band at y_0 is the *front* rim; the other one is the *back* rim. For any other band b, the rim adjacent to its parent is the *front* rim, and the other is the *back* rim. In Figure 2a, for example, the front rim of b_8 is at y_1. A child is a *front child* (*back child*) if it is adjacent to the front (back) rim of its parent. In Figure 2b, thin (thick) arcs connect a parent to its front (back) children. (Note the sense of front and back used here to distinguish rims and children is relative to the parent band, rather than based on $\pm y$.)

In the following we describe the recursive unfolding algorithm. We begin by establishing that there exists a simple spiraling path ξ on the surface of O that starts and ends on the front rim of the root band and winds around each band in T_U at least once. When this path is "thickened," it covers the band faces and unfolds into a horizontal staircase-like strip to which front and back faces of O can be attached vertically. We describe ξ recursively, starting with the base case in which O consists of a single box and thus the partition π leaves a single band.

2.1. Single Box Spiral Path

Let O be a box with band b. We use the following notation (see Figure 3a): A, B, C, and D are top, right, bottom and left faces of O (these faces belong to b); E and F are back and front faces of O; s and t are *entering* and *exiting* points on the top edge of b's front rim.

The main idea is to start at s, spiral forward[1] around band faces A, B, C and D, cross the back face E to reverse the direction of the spiral, then spiral back to t. See Figure 3a for an example, where mirror views are provided for faces that cannot be viewed directly. We refer to the forward spiral (incident to entering point s) as the *entering* spiral, and the backward spiral (incident to exiting point t) as the *exiting* spiral. Spiral ξ is the concatenation of the entering spiral, the

[1] By spiraling forward we mean spiraling towards the back rim

back face strip (K_0 in Figure 3a), and the exiting spiral. It can be unfolded flat and laid out horizontally in a plane. See Figure 3b.

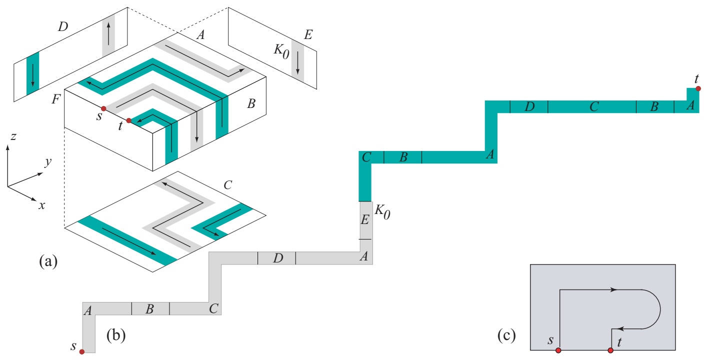

Fig. 3. (a) Spiral path ξ with configuration R_{st} (b) ξ flattened. (c) 2D representation.

We distinguish four variations of this spiraling path, which differ in how they enter and exit the band: the entering spiral is heading away from s either to the right (cw) or to the left (ccw), and t is either to the left or to the right of s on the front edge. Directions left, right, cw, and ccw are defined w.r.t a viewer at $y = -\infty$. We will use the notation R_{st}, R_{ts}, L_{st} and L_{ts}, to identify the four possible entering/exiting configurations. Here the first letter (R or L) indicates the direction (Right or Left) the entering spiral is heading as it moves away from s, and the subscripts indicate the position of s relative to t, as viewed from $y = -\infty$. We use the symbol R_{-} when the relative position of s and t is irrelevant to the discussion; i.e., R_{-} denotes either R_{st} or R_{ts}, and same for L_{-}. Thus there are four base cases, one for each entering/exiting configuration. Figure 3a illustrates R_{st}; R_{ts}, L_{st}, and L_{ts} are analogous.

Three dimensional illustrations of ξ, such as Figure 3a, are impractical for all but the smallest examples. To be able to illustrate more complex unfoldings, we define a 2D representation that is illustrated in Figure 3c for the R_{st} base case. Note that the 2D representation captures the direction of the entering spiral and the relative position of s and t. The entrance is connected by an arc to the exit, symbolizing the forward spiral reversing its direction using a back face strip (K_0 in Figure 3a). Representations for the other three base cases are analogous and can be identified in Figures 4 and 5.

2.2. Recursive Structure

In general, a band b has children adjacent along its front and back rims. The spiral path ξ for the subtree rooted at b begins and ends at points s and t on the top edge of b's front rim. The entering and exiting spirals of b conform to one of the four entering/exiting configurations R_{st}, R_{ts}, L_{st} or L_{ts}.

We describe ξ at a high level first. Once b's entering spiral leaves s, it follows an alternating path (switching between spiraling cw and ccw, as in Figure 4) to reach each of the front children of b and spiral around them recursively. The path on b alternates between cw and ccw because spiraling around a child reverses the direction of b's spiral. After visiting the front children, b's entering spiral cycles around b to its back rim, where it follows a second alternating path (see Figure 5) to reach each of the back children and spiral around them recursively. After visiting the last back child, b's exiting spiral returns to t, tracking the path taken by the entering spiral, but in reverse direction. This final reverse spiral will revisit nodes/bands already visited on the forward pass, and the recursive structure will imply that some nodes/bands will be revisited many times before the spiral returns to t. We defer discussion of this consequence of the algorithm to Section 4.

2.2.1. Alternating Paths for Labeling Children A preorder traversal of T_U assigns each band an entering/exiting configuration label (R_{st}, R_{ts}, L_{st}, or L_{ts}). Although any label would serve for the root box of T_U, for definitiveness we label it R_{ts}. We also pick an entering and an exiting point on top of the root's front rim, with the exiting point to the left of the entering point, which is consistent with its R_{ts} label. We provide algorithms for labeling the front and back children of a labeled parent b, which get applied when b is visited during the traversal. These rules are described in terms of two alternating paths that b's entering spiral takes to reach every front and back child.

We begin with the alternating path for labeling b's front children. See Algorithm 1. Observe that in Step 2(b) we must reverse the current direction because the spiral exits a child box heading in the opposite direction from which it entered. This forces the left/right alternation between the front children of b. Figure 4 shows an example in which b has five front children and an R-type configuration. The children are visited in the order b_1, b_2, b_3, b_4 and b_5; the dashed lines correspond to walking around side and bottom faces of b, to reach an unlabeled child from the top of b. The configuration assigned to each child by the labeling algorithm is shown within parentheses.

Algorithm 1 LABEL-FRONT-CHILDREN(b) (see Figure 4)

```
1. Set current position to b's entering point $s$. Set current direction to that of b's
   entering spiral: rightward, if $b$ has an $R$-label, and leftward if an $L$-label.
2. **while** $b$ has unlabeled front children **do**
   (a) From the current position, walk in the current direction along the front rim of
       $b$, until (i) an unlabeled front child $b_i$ is encountered, and (ii) current position
       is on top of $b$.
   (b) Assign to $b_i$ label $R_{st}$ ($L_{ts}$) if walking rightward (leftward). Select points $s_i$ and $t_i$
       on the top line segment at the intersection between $b$ and $b_i$, in a relative position
       consistent with the label ($R_{st}$ or $L_{ts}$) of $b_i$. Reverse the current direction.
```

After labeling all front children of b, we label the back children of b using a similar scheme. See Algorithm 2. Note that the algorithm ensures that the relative position of s_i and t_i for the back child last labeled is the same as for parent b. For example, b in Figure 5 has five back children, and is a $\_ts$ unfolding

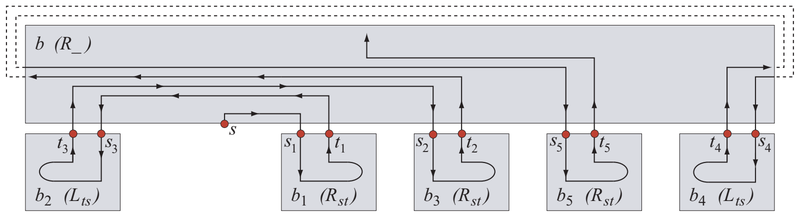

Fig. 4. Alternating path for labeling b's front children.

configuration. The back children get visited in the order $b_6, b_7, b_8, b_9, b_{10}$. The unfolding label $\_ts$ assigned to each back child is shown within parentheses. Observe that the unfolding label $\_ts$ for b_{10} (the back child last labeled) is consistent with the unfolding label $\_ts$ of its parent. Also observe that the nesting of the L/R alternation is inside-out (outside-in) for front (back) children (cf. Figures 4 and 5).

Algorithm 2 LABEL-BACK-CHILDREN(b) (see Figure 5)

```
1. Set current position to $s$, if $b$ has no front children; otherwise, set current position
   to the exiting point of the front child last labeled.
2. Set current direction to the direction of $b$'s entering spiral, if $b$ has no front children;
   otherwise, set current direction to the direction of the exiting spiral of the front child
   $b_i$ of $b$ last labeled - leftward if $b_i$ has an $R$-label, rightward if $b_i$ has an $L$-label.
3. **while** $b$ has unlabeled back children **do**
   (a) If the current direction is leftward (rightward), then walk leftward (rightward)
       from the current position, until the leftmost (rightmost) unlabeled back child
       $b_i$ is encountered.
   (b) If $b_i$ is not the last unlabeled back child, then assign to $b_i$ label $L_{st}$ ($R_{ts}$), if the
       current direction is leftward (rightward).
   (c) If $b_i$ is the last unlabeled back child, assign to $b_i$ a label with the same ordering
       of $s$ and $t$ as for $b$. Specifically, if $b$ has a $\_st$ label and $b_i$ is entered while heading
       leftward (rightward), assign to $b_i$ label $L_{st}$ ($R_{st}$); if $b$ has a $\_ts$ label and $b_i$ is
       entered while heading leftward (rightward), assign to $b_i$ label $L_{ts}$ ($R_{ts}$).
   (d) Select points $s_i$ and $t_i$ on the top line segment at the intersection of $b$ and $b_i$, in
       a relative position consistent with the label of $b_i$. Reverse the current direction.
```

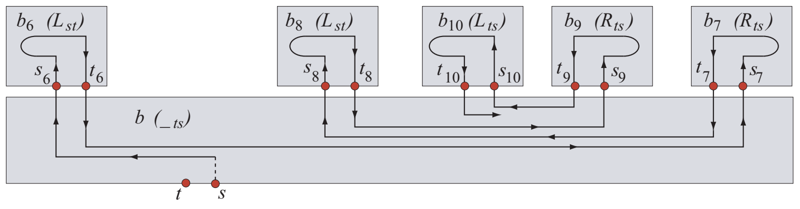

Fig. 5. Alternating path for labeling b's back children.

The path exiting b_{10} must now return to the exiting point t of parent b, and to do so, because it is deeply nested in the alternating paths, it must follow the entire path between the entering point of b and the entering point of b_{10}, but in reverse direction. We return to this in the next section.

2.2.2. Recursive Spiral Paths The spiral path for a subtree rooted at band b is computed recursively, cf. Algorithm 3. Lemma 1 establishes the correctness of this algorithm.

Algorithm 3 SPIRAL-PATH(b) (see Figure 6)

```
1. If $b$ has no children, follow the appropriate base-case spiral path and return.
2. If $b$ has no front children, skip to Step 4.
3. while ($b$ has unvisited front children) do
   2.1 Follow front alternating path to entering point of next front child $b_i$ of $b$.
   2.2 SPIRAL-PATH($b_i$).
4. Complete a cycle around $b$ and proceed to the back rim of $b$.
5. If $b$ has no back children, reverse spiral using a back face strip (analogous to $K_0$ in
   Fig 3) and skip to Step 7.
6. while ($b$ has unvisited back children) do
   4.1 Follow back alternating path to entering point of next back child $b_i$ of $b$.
   4.2 SPIRAL-PATH($b_i$).
7. Retrace the entering spiral for $b$ back to the exiting point of $b$.
```

Lemma 1. *For any unfolding tree T_U rooted at a node with entering point s and exiting point t, there exists a simple spiral path ξ such that (i) ξ starts at s, cycles around each band in T_U at least once, and returns to t heading in the reverse direction (ii) for each node b in T_U, ξ is consistent with the entering/exiting configuration for b, and (iii) ξ unfolds flat horizontally, with s on the far left and t on the far right.*

Proof. The proof is by induction on the depth of T_U. The base case corresponds to a tree with a single node b (of depth 0), and is established by Figure 3 for configuration R_{st}, and can be established analogously for R_{ts}, L_{st}, and L_{ts}.

Assume that the lemma holds for unfolding trees of depth d and less, and consider an unfolding tree T_U of depth $d+1$ rooted at b. Assume b has an R_{ts} entering/exiting configuration; cases for R_{st}, L_{ts} and L_{st} are similar. Consider the general case when b has both front and back children. The spiral ξ starts at s and follows the front alternating path (e.g. Figure 4) to reach each front child b_i. By the inductive hypothesis there exists a spiral path $\xi(b_i)$ from s_i to t_i for b_i's subtree. From t_i, ξ continues on to the next front child. After exiting the last front child, ξ makes one complete cycle around b, and then follows the back alternating path (e.g. Figure 5) to reach each back child. Again the inductive hypothesis gives us a spiral path for each back child's subtree.

The portion of ξ from s to the entering point of the back child last visited is b's entering spiral. Once ξ leaves the last back child, it must return to the exiting point t of b, and it does so by tracking b's entering spiral in the reverse direction.

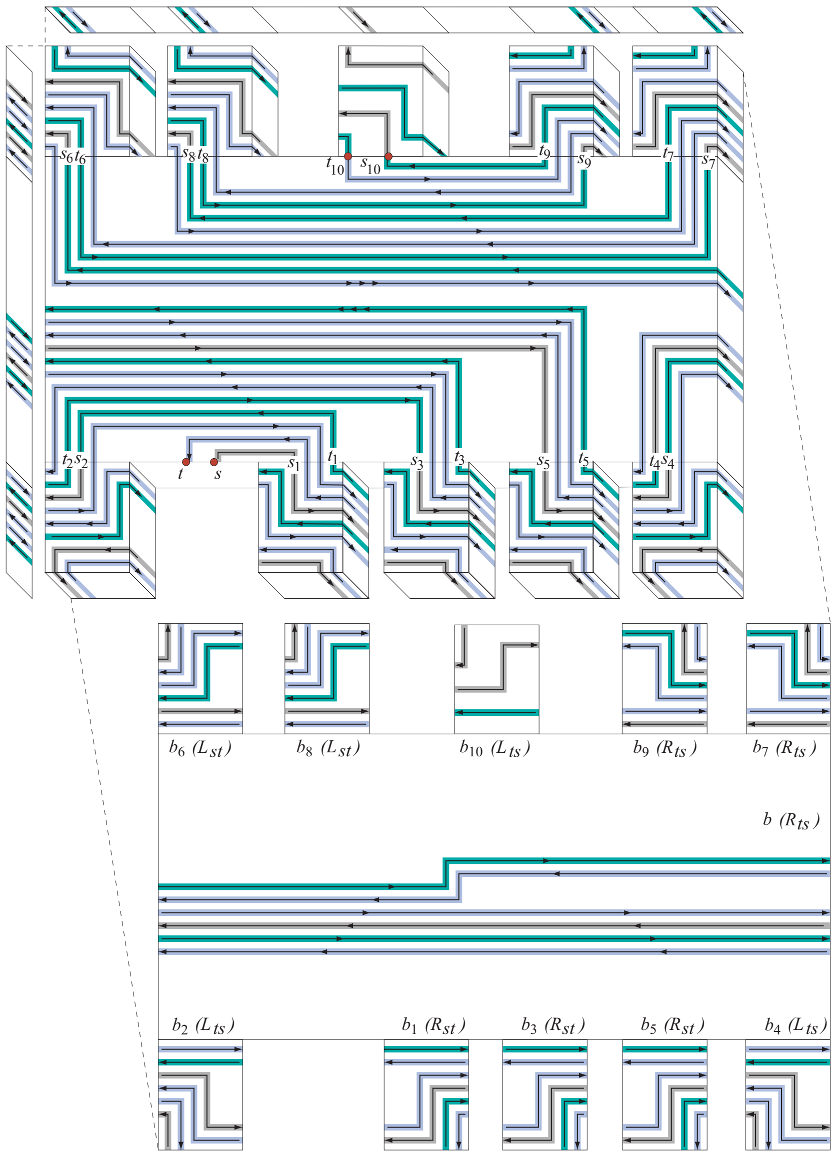

Fig. 6. Spiral ξ for a complete example, with mirror views for side and bottom faces.

This portion of ξ is b's exiting spiral. Figure 6 illustrates ξ in its entirety for a depth-1 unfolding tree. The spiral for each child corresponds to one of the four base cases. It is straightforward to prove by induction that ξ satisfies the three conditions stated in the lemma. For details, see [7]. $\square$

2.3. Recursive Spiral Path Example

To reinforce our recursive spiraling ideas, we provide the 2D representation of an unfolding for a more complex example, illustrated in Figure 7. Observe that the cycle that ξ makes around each band between visiting the front and back children is not captured by the 2D representation, therefore we omit mentioning it in this section.

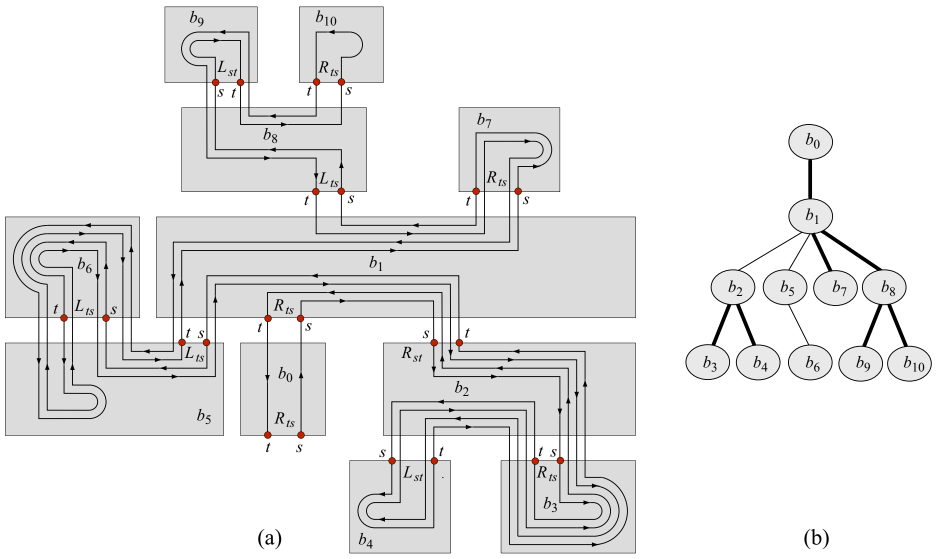

Fig. 7. A recursive unfolding example: (a) 2D representation. (b) Unfolding tree.

The algorithm begins by assigning an R_{ts} label to b_0, and then uses the algorithms from Section 2.2.1 to assign labels to the other bands, as marked in Figure 7. The spiral ξ then starts at b_0 and cycles forward to its only back child b_1. After entering b_1, it proceeds along the front alternating path to front child b_2. Since b_2 has no front children ξ proceeds to the back of b_2 to visit its back children b_3 and b_4, in this order. Once it exits the last back child b_4, ξ tracks the entering spiral for b_2 in reverse back to t on the front rim of b_2, passing through b_3 again on the way. After exiting b_2, ξ follows b_1's front alternating path to b_5, follows b_5's front alternating path to b_6, and then proceeds to the back of b_5. Since b_5 has no back children, ξ reverses direction using a back face strip and begins tracking the path back to b_1, thus visiting b_6 again. ξ reenters b_1 and moves along its back alternating path to visit b_7 and then on to b_8. Note that b_8 is the last back child of b_1 to be visited, therefore the spiral between the entering point of b_1 and the entering point of b_8 is the entering spiral of b_1. Between the entering and exiting points of b_8, ξ visits back children b_9 and b_{10}, then b_9 again on its way back to the front of b_8. It then tracks b_1's entering spiral in reverse to the front rim of b_1. Finally, it tracks b_0's entering spiral to the front rim of b_0.

2.4. Thickening ξ

Spiral ξ established in Section 2.2.2 can be thickened in the y direction so that it entirely covers each band. This results in a vertically thicker unfolded strip. See Figure 8 for a two-band example. Since the unfolded ξ is monotonic in the horizontal direction, thickening it vertically cannot result in overlap.

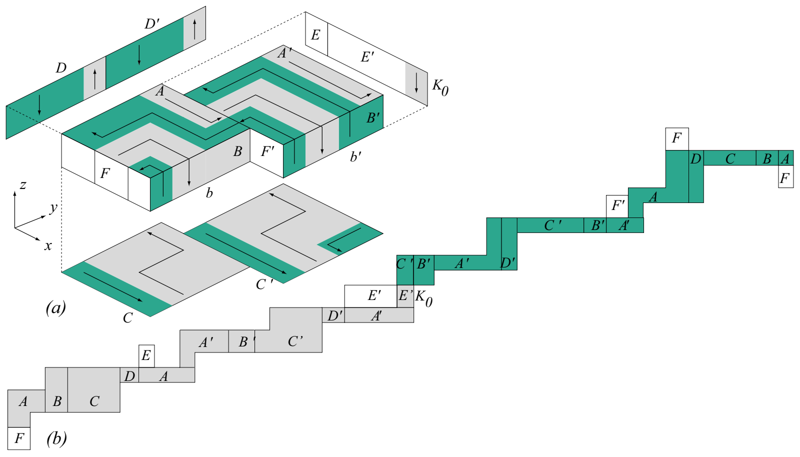

Fig. 8. (a) Thickened ξ. (b) ξ unfolded with front and back face pieces attached.

2.5. Attaching Front and Back Faces

Finally, we "hang" the front and back faces of O from the thickened ξ in a manner similar to that done in [8], as follows. We first partition the front and back faces of O by imagining the bands' top rim edges illuminating downward lightrays in these faces. The illuminated pieces can then be attached above and below ξ along the corresponding illuminating rim segments. For an example, see Figure 8 showing a two band shape and its unfolding. Here the front face of band b is partitioned into three pieces which are hung from their corresponding rim segments in the unfolding. Faces E, F' and E' are hung similarly. Observe that edge overlap mentioned in Section 1 occurs between face F' and a section of A in the unfolding.

This completes the unfolding process, which we summarize in Algorithm 4.

3. Unfolding All Orthogonal Polyhedra

The unfolding algorithm described for extrusions generalizes to unfolding all orthogonal polyhedra. Let O be a genus-zero orthogonal polyhedron. The surface of O is simply connected, which means that any closed curve on the surface can

Algorithm 4 UNFOLD-EXTRUSION(O)

```
1. Partition $O$ into bands (Section 1).
2. Compute unfolding tree $T_U$ with root band $b_0$.
3. For each band $b$ encountered in a preorder traversal of $T_U$
   3.1 LABEL-FRONT-CHILDREN($b$).
   3.2 LABEL-BACK-CHILDREN($b$) (Section 2.2.1).
4. Determine $\xi$ = SPIRAL-PATH($b_0$) (Sections 2.1, 2.2.2).
5. Thicken $\xi$ (Section 2.4) and hang front and back faces (Section 2.5).
```

be continuously contracted on the surface to a point. We will use this characterization in our proofs.

Unlike the extrusions of Section 2, general orthogonal polyhedra may have dents, as defined in Section 1. As we observed in [8], dents may be treated exactly the same as protrusions with respect to unfolding by conceptually “popping” them out to become protrusions. (This popping-out is conceptual only, for it could produce self-intersecting objects.) Henceforth, we will describe only protrusions in our algorithm, with the understanding that nothing changes for dents.

3.1. Determining Connecting z-beams and Computing Unfolding Tree T_U

We start by partitioning O into bands with planes $Y_0, Y_1, ..., Y_i, ...$ through each vertex. Define a *z-beam* to be a vertical rectangle on the surface of O of nonzero width connecting two band rims. In the degenerate case, a z-beam has height zero and connects two rims along a section where they coincide. We say that two bands b_1 and b_2 are *z-visible* if there exists a z-beam connecting an edge of b_1 to an edge of b_2.

Lemma 2. *All z-beams between two z-visible bands lie in one Y_i plane.*

Proof. Suppose to the contrary that bands b_1 and b_2 are connected by beams in both Y_i and Y_{i+1}, i.e., both rims of both bands are connected by z-beams. Then we can construct a closed curve C on the surface of O from b_1, following the beam on Y_i to b_2, and following the beam on Y_{i+1} back to b_1. Now let E be a closed curve just exterior to, say b_1, parallel to and between Y_i and Y_{i+1}. Then E and C are interlinked. This means that C cannot be contracted to a point, contradicting the genus-zero assumption. □

Thus all z-beams between two z-visible bands are in this sense equivalent. We select one z-beam of minimal (vertical) length to represent this equivalence class.

Let G be the graph that contains a node for each band of O and an arc for each pair of z-visible bands. It easily follows from the connectedness of the surface of O that G is connected. Let the unfolding tree T_U be any spanning tree of G, with the root selected arbitrarily from among all bands adjacent to Y_0.

As defined in Section 1, the rim of the root node/band at y_0 is called its *front* rim; the other is its *back* rim. For any other band b, we provide definitions equivalent to the ones in Section 2, only this time in terms of connecting z-beams: the *front* rim of b is the one to which the (representative) z-beam to its parent is attached; and the other rim of b is its *back* rim (Lemma 2 guarantees that this definition is unambiguous.) A child is a *front child* (*back child*) if its z-beam connects to the front (back) rim of its parent. We call the region of the Y-plane enclosed by a band's back rim its *back face*, and we say that the back face is *exposed* if it is a face of O.

The following lemma establishes that bands with no back children in T_U have exposed back faces. This is important to our unfolding algorithm because it employs strips (such as K_0 in Figure 3) from exposed back faces to turn the spiral around. Figure 9a shows that this lemma does not hold true for arbitrary genus-one objects: b_2 has no back children (cf. unfolding tree to its right), however its back face is not exposed.

Lemma 3. *The back face of every band in T_U with no back children is exposed.*

Proof. We show by contradiction that a band with no back children in T_U must have no bands resting on its back face, and thus its back face is fully exposed. We do this by proving the existence of a simple, closed surface curve that cannot be contracted to a point, thus violating our genus-zero assumption.

Let p and q be two arbitrary band points on O, and let $(b_1, b_2, ..., b_k)$ be the path in T_U between the band b_1 containing p and the band b_k containing q. We begin by establishing that p and q are connected by a simple surface curve that follows the path in T_U between b_1 and b_k band. Let $z_1, z_2, ..., z_{k-1}$ be the z-beams connecting pairs of adjacent bands along this path. From p, the surface curve moves around b_1 until it meets z_1, then along z_1 to b_2. In a similar manner the curve moves from b_2 to b_3 and so on, until it reaches b_k. Once on b_k, the curve moves around b_k to q. See Figure 9b.

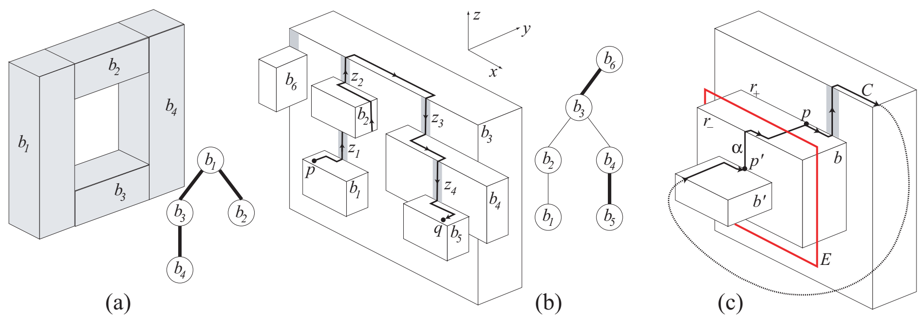

Fig. 9. (a) A genus-1 object: b_2 has no back children (b) A surface curve corresponding to a five-band path in T_U (on the right). (c) Two interlinked closed curves, C on the surface of O and E exterior.

Now suppose for the sake of contradiction that there is a band b with no back children whose back face is not exposed. Let r_- be the back rim of b and r_+ the front rim of b (cf. Figure 9c). If the back face of b is not exposed, there must be some vertical ray α that extends from r_- to a point p' on some other band b', as shown in Figure 9c. (Note that if the rims of b and b' coincide, then the ray can degenerate to a point.)

Let p be a point on the front rim r_+ of b. We established above the existence of a particular surface curve C from p to p' corresponding to the path between b and b' in T_U. Because b has no back children in T_U, C moves from r_+ to the parent of b or a front child and never returns to b again. We can now extend C to a simple closed curve by moving from p to r_- and then along ray α to p' (see Figure 9c.) Now consider a second closed curve E exterior to O that cycles around band b. Curves C and E are interlinked, meaning that C cannot be contracted to a point. This contradicts the genus-zero assumption. □

3.2. Unfolding Algorithm

Once T_U is determined, the algorithm that unfolds all orthogonal polyhedra is very similar to the algorithm for unfolding extrusions. We summarize it here, noting the differences.

The unfolding starts by assigning an entering/exiting configuration to each band in T_U by following alternating paths on each band to reach its children's connecting z-beams. The alternating paths are similar to those in Section 2.2.1, but substituting ccw and cw for directions left and right, walking until a z-beam of an unlabeled child is encountered (as opposed to walking until the top edge of a child is encountered), and placing the entering/exiting points where the connecting z-beam touches the parent's rim. Note that, unlike in the case for extrusions, a z-beam may connect a child to a bottom edge of its parent (see b_3 in Figure 10a), forcing the alternating path to go around to the bottom face of the parent until it reaches the child.

The algorithm then determines a spiral ξ with the properties listed in Lemma 1. This is done as in Section 2.2.2, but here ξ follows the alternating paths to each child's z-beam, travels along the vertical z-beam to the child band, recursively spirals around the child, and then travels along the z-beam back to the parent. These z-beams unfold vertically in the plane, merely making some steps taller in the unfolded staircase. For bands with no back children, ξ reverses direction using a strip from the band's back face. Lemma 3 establishes that the back faces of such bands are exposed, and hence a strip is available. Any vertical strip extending from a top to a bottom edge of the back face may be used. See Figure 10 for a complete example; here the spiral begins on band b_0 cycling cw.

Finally, ξ is thickened as in Section 2.4, and then front and back faces of O are partitioned and attached according to the illumination model described in Section 2.5, with one modification—here both top *and* bottom edges of each rim illuminate downward lightrays. The front and back face pieces are attached to their illuminating rim segments. (See Figure 10b.) Bottom edges must illuminate light because lower bands may block rays from higher bands (which cannot occur with extrusions). This method is guaranteed to illuminate all front and back faces since a ray shot upward from any front or back face point will hit a top or bottom edge of a rim before leaving the surface of O. The rim it hits is the one that illuminates it and the one to which its piece is attached.

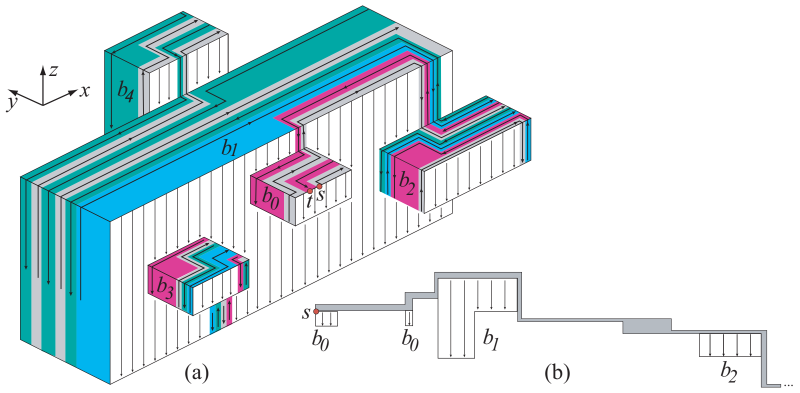

Fig. 10. (a) Four-block example (scale reduced for space considerations); $b_0, b_2 (b_3)$ hang from the top (bottom) of b_1. (b) Prefix of unfolding (not to same scale).

4. Worst Case

The thinness of the spiral is determined by the number of parallel paths on any face, which we call the *path density* on that face. If the maximum density is k, then the thickened spiral path can be at most $1/k$-th of the face width (y-extent). We say a band b_i is *visited* each time the spiral enters the band, alternates back and forth between its children, turns around, and alternates between its children in reverse order, and finally exits b_i. If there are m parallel paths on a face after the first visit, then after v visits there are vm parallel paths, since each subsequent visit tracks alongside a path laid down during an earlier visit. For example, Figure 3 shows a single box visited once with 4 parallel paths on its top face. Figure 6 shows that except for the turnaround box (b_{10}), boxes at depth $d = 1$ are visited twice, doubling their path densities to $8 = 2 \times 4$.

We now construct an example that has path density $2^{\Omega(n)}$, where n is the number of vertices of the polyhedron. We use a skewed sequence of extrusions (viewed from $+z$) in Figure 11. In each case, the spiral starts on the bottom box heading to the right. Marked on each leaf box is the number of times the box is visited. One of the children at depth d (shaded in the figure) is visited 2^d times and has a path density of $4 \cdot 2^d$. A depth-d tree of this structure contains $2d+1$ boxes total, and so can be realized by a polyhedron with $n = 8(2d+1)$ vertices. Thus $d = \Omega(n)$, and the shaded child has a path density of $2^{\Omega(n)}$. We conclude that the spiral path may need to be as thin as $\varepsilon = 1/2^{\Omega(n)}$ times the smallest y-extent of any face of the polyhedron.

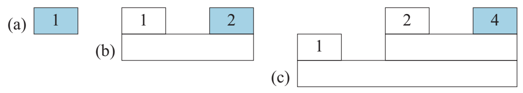

Fig. 11. One back child has path density $4 \cdot 2^d$.

To establish a density upper bound of $2^{O(n)}$, observe that each time a band is visited, its children are visited at most twice, and therefore 2^d bounds the number of visits for a band at depth d. It can be that the most dense band is not a leaf, but a band having many children. As Figure 6 makes clear, the number of parallel paths, m, laid out on each visit depends on the number of children a band has, due to the alternation back and forth to each child. Noting that both m and d are $O(n)$, we can conclude that the path density is bounded by $m2^d = O(n)2^{O(n)}$, which is still $2^{O(n)}$.

Theorem 1. *The path density is $2^{\Theta(n)}$, and so $\varepsilon = 1/2^{\Theta(n)}$ in the worst case.*

5. Conclusion

We have established that every orthogonal polyhedron of genus zero may be unfolded. We believe that our algorithm can be extended to handle orthogonal polyhedra with genus ≥ 1. One idea is to treat holes as being blocked by virtual membranes, unfold according to our genus-0 algorithm, and then compensate for the virtual faces. However, a number of details in such an algorithm would need careful handling. A second extension of our algorithm would be to construct a $k \times k$-refined grid unfolding, for constant k. Although our algorithm fundamentally relies on ε-thin strips, a mix of our current unfolding techniques with the ones employed in [6] to reverse the direction of the unfolding, may help achieve this extension.

Finally, our spiraling technique so relies on the orthogonal structure of the polyhedra that it seems difficult to use it in resolving the open problem of whether every polyhedron may be unfolded.

Acknowledgment. We are grateful for insightful comments of a referee.

References

1. Biedl, T., Demaine, E., Demaine, M., Lubiw, A., O'Rourke, J., Overmars, M., Robbins, S., Whitesides, S.: Unfolding Some Classes of Orthogonal Polyhedra. In: *Proc. 10th Canad. Conf. Comput. Geom.,* 70–71 (1998)
2. Demaine, E.D., Eppstein, D., Erickson, J., Hart, G.W., O'Rourke, J.: Vertex-Unfoldings of Simplicial Manifolds. In: A. Bezdek, editor: Discrete Geometry, 215–228, New York: Marcel Dekker (2003)
3. Demaine, E.D., Iacono, J., Langerman, S.: Grid Vertex-Unfolding of Orthostacks. In: *Proc. Japan Conf. Discrete Comp. Geom.*, Lecture Notes in Comput. Sci. **3742**, 76–82, Springer (2004)

4. Demaine, E.D., O'Rourke, J.: A Survey of Folding and Unfolding in Computational Geometry. In: J. E. Goodman, J. Pach, and E. Welzl, editors: Combinatorial and Computational Geometry, 167–211, Cambridge University Press (2005)
5. Demaine, E.D., O'Rourke, J.: Open Problems from CCCG 2004. In: *Proc. 17th Canad. Conf. Comput. Geom.*, 303–306 (2005)
6. Damian, M., Flatland, R., O'Rourke, J.: Unfolding Manhattan Towers. In: *Proc. 17th Canad. Conf. Comput. Geom.*, 204–207 (2005)
7. Damian, M., Flatland, R., O'Rourke, J.: Epsilon-Unfolding Orthogonal Polyhedra. Technical Report 082, Dept. Comput. Sci., Smith College, Northampton, MA, USA. `http://arXiv.org/abs/cs.CG/0602095/` (2006)
8. Damian, M., Flatland, R., O'Rourke, J.: Grid Vertex-Unfolding Orthogonal Polyhedra. In: *Proc. 23rd Symp. on Theoretical Aspects of Comp. Sci.*, Lecture Notes in Comput. Sci. **3884**, 264–276, Springer (2006)