What Does Digital Straightness Tell About Digital Convexity ?

Tristan Roussillon*[1], Laure Tougne[1], and Isabelle Sivignon[2]

[1] Université de Lyon,
Université Lyon 2, LIRIS, UMR5205, F-69676, FRANCE
{tristan.roussillon, laure.tougne}@liris.cnrs.fr
[2] Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205, F-69622, FRANCE
isabelle.sivignon@liris.cnrs.fr

Abstract. The paper studies local convexity properties of parts of digital boundaries. An online and linear-time algorithm is introduced for the decomposition of a digital boundary into convex and concave parts. In addition, other data are computed at the same time without any extra cost: the hull of each convex or concave part as well as the Bezout points of each edge of those hulls. The proposed algorithm involves well-understood algorithms: adding a point to the front or removing a point from the back of a digital straight segment and computing the set of maximal segments. The output of the algorithm is useful either for a polygonal representation of digital boundaries or for a segmentation into circular arcs.

1 Introduction

The paper studies local convexity properties of parts of digital boundaries. As shown in [Eck01], the convexity of a digital boundary cannot be decided locally (where locally means in the 8-neighbourhood). Considering this fact, the following question has been raised in [EDR04]: how far one can decide whether a part of a digital boundary is convex or not by a method that is *as local as possible*? Our answer is that a good neighbourhood for checking convexity is given by a segment that cannot be extended either at the front or at the back.

An online and linear-time algorithm is introduced for the decomposition of a digital boundary into convex and concave parts. The proposed algorithm uses well-understood algorithms: adding a point to the front [DRR95] or removing a point from the back [LVdV07] of a digital straight segment. The core of the algorithm is similar to the one that computes the set of maximal segments [FT99,LVdV07]. Hence, one single scan of a digital boundary by a window corresponding to a digital straight segment is sufficient to decompose the digital boundary into convex and concave parts. Our algorithm is on-line (contrary to [Fes05,DRDR05]) and leads to a unique decomposition (contrary to [DRRRD03]).

* Author supported by a grant from the DGA

Moreover, other data are computed simultaneously. During the scan, the hull of each convex or concave part as well as the Bezout points of each edge of each hull are computed (we call *hull* a partial convex hull whose formal definition is given in Section 3.1). To do this, some operations are added to the maximal segments computation when the first and last leaning points of the current digital straight segment merge or split.

The link between the leaning points of the maximal segments and the hull of convex or concave parts has been investigated either for local estimators [dVLF07] or for faithful polygonalizations [EDR04,DRDR06]. In this paper, the link between the hull of convex or concave parts and the leaning points of digital straight segments that are not necessarily maximal is also studied.

In Section 2, definitions of digital boundary, digital straight segment, leaning points and Bezout points are recalled in detail. The main contribution consists of Proposition 2 and Corollary 1 proved in Section 3.2. These propositions yield to Algorithm 2, which is an extended version of Algorithm 1. Applications of these algorithms are discussed in Section 4.

2 Preliminaries

2.1 Digital Object and Digital Boundary

A binary image I is viewed as a subset of points of $\mathbb{Z}^2$ that are located inside a rectangle of size $M \times N$. A digital object $O \in I$ is a 4-connected subset of $\mathbb{Z}^2$, without hole (Fig. 1.a). Its complementary set $\bar{O} = I \backslash O$ is the so-called background. The digital boundary B of O is defined as the 8-connected clockwise-oriented list of the digital points having at least one 4-neighbour in $\bar{O}$ (Fig. 1.b).

Each point of B is numbered according to its position in the list. The starting point, which is arbitrarily chosen, is denoted by B_0 and any arbitrary point of the list is denoted by B_k. A part $(B_i B_j)$ of B is the list of points that are ordered increasingly from index i to j (Fig. 1.c).

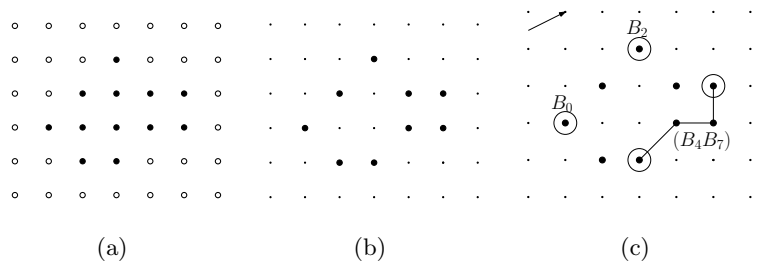


Fig. 1. A digital object depicted with black disks (a). Its digital boundary (b). Notation used (c)

2.2 Digital Straight Line

Definition 1 (Digital straight line [Rev91]) *The set of digital points* (x, y) *verifying* $\mu \leq ax - by < \mu + max(|a|, |b|)$ *belongs to the digital straight line (DSL)* $\mathcal{D}(a, b, \mu)$ *with slope* $\frac{a}{b}$ *and lower bound* μ *(with* a*,* b*,* μ *being integer such that* $gcd(a, b) = 1$*).*

The quantity $ax - by$, which is called the *remainder*, measures the distance between (x, y) and $\mathcal{D}$. Table 1 clusters the digital points of $\mathbb{Z}^2$ into seven groups according to their position with respect to $\mathcal{D}$. Note that merging the last two lines of Table 1 gives the two inequalities of definition 1.

position	on the left	on the right
strongly exterior	$ax - by < \mu - 1$	$ax - by > \mu + max(\|a\|, \|b\|)$
weakly exterior	$ax - by = \mu - 1$	$ax - by = \mu + max(\|a\|, \|b\|)$
weakly interior	$ax - by = \mu$	$ax - by = \mu + max(\|a\|, \|b\|) - 1$
strongly interior	$\mu < ax - by < \mu + max(\|a\|, \|b\|) - 1$	

Table 1. The digital points of $\mathbb{Z}^2$ are divided into seven groups according to their position with respect to the DSL $\mathcal{D}(a, b, \mu)$

Thanks to the vocabulary introduced in Table 1, two special kinds of points are easily defined:

Definition 2 (Leaning points and Bezout points (Fig. 2.a)) *The* leaning points *(resp.* Bezout points*) of a DSL* $\mathcal{D}$ *are defined as the points that are weakly* interior *(resp.* exterior*) to* $\mathcal{D}$*.*

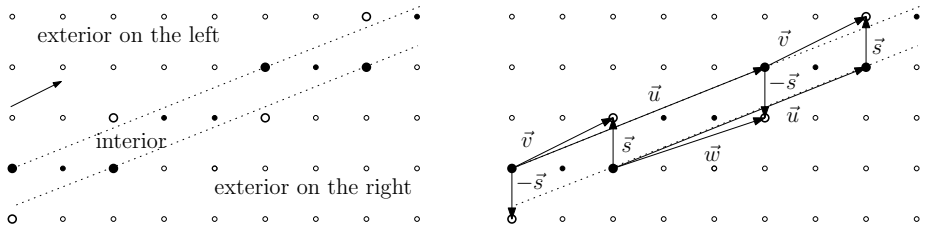
The difference between two consecutive leaning points both located on the left or right of $\mathcal{D}$ is the vector $\vec{u} = (b, a)$ (Fig. 2.b).

The Bezout points are closely related to the leaning points. Indeed, the Bezout points that are on the left or right of $\mathcal{D}$ may be computed from the leaning points of the same side thanks to the well-known Bezout's identity: vector $\vec{v}$ (resp. $\vec{w}$) of Fig. 2.b is such that $det(\vec{u}, \vec{v})$ equals -1 (resp. $det(\vec{u}, \vec{w})$ equals 1) (Fig. 2.b). Moreover, the leaning points on the left or right of $\mathcal{D}$ maps to the Bezout points of the opposite side when shifted by a vector $\vec{s}$ that depends on the slope of $\mathcal{D}$. In the first octant, $\vec{s} = (0, 1)$ (Fig. 2.b).

2.3 Digital Straight Segment

Definition 3 (Digital straight segment) *A part* $(B_i B_j)$ *of* B *is a* digital straight segment *(DSS) if and only if there exists a DSL containing it.*

There are infinitely many DSL containing a DSS $(B_i B_j)$. However, there is always one DSL that is *strictly bounding* for $(B_i B_j)$.



(a) (b)

Fig. 2. The set of black disks lying between the two parallel dotted lines defines the DSL $\mathcal{D}(2, 5, 0)$. The large black and white disks depict the leaning (weakly interior) and Bezout (weakly exterior) points of $\mathcal{D}$. In (b), vectors $\vec{v}$, $\vec{w}$ and $\vec{s}$ show that the Bezout points are closely related to the leaning points

Definition 4 *A DSL is strictly bounding for a DSS* (B_iB_j) *if it has at least three leaning points belonging to* (B_iB_j)*. Moreover, at least one of them is a leaning point that is on the left of the strictly bounding DSL and at least one of them is a leaning point that is on the right of the strictly bounding DSL.*

The recognition algorithm of Debled and Reveillès [DRR95] returns the parameters a, b, μ of the strictly bounding DSL containing a DSS.

When speaking about a DSS (its slope for instance), we automatically refer to its strictly bounding DSL. Therefore, the exterior and interior points of a DSS (B_iB_j) are defined as the exterior and interior points of the strictly bounding DSL containing (B_iB_j). However, among all the leaning points of a DSS (B_iB_j), only those contained in the DSS, with an index ranging from i to j, are retained. Thus, the first (resp. last) leaning point is defined as the one with a minimal (resp. maximal) index.

Definition 5 (Maximal segment) *A DSS* (B_iB_j) *that cannot be extended at the front (resp. at the back),* i.e. (B_iB_{j+1}) *(resp.* $(B_{i-1}B_j)$*) is not a DSS, is said maximal at the front (resp. at the back). Moreover, a DSS that is both maximal at the front and maximal at the back is a maximal segment.*

There exist two algorithms to add [DRR95] or remove [LVdV07] a point at one extremity of a DSS in constant time. Thanks to these two algorithms, the computation of the whole set of maximal segments of a digital boundary is done in linear time [FT99,LVdV07].

The set of maximal segments contains all DSS and all DSS segmentations (Fig. 8.a), one of which has the minimal number of segments [FT05]. Estimations of length [CK04], tangent [FT99,LVdV07] or curvature [FT99,CMT01] may be derived from this set. Moreover, convex and convave parts are given by the slopes of the maximal segments [Fes05,DRDR05].

3 Local Convexity and DSS

3.1 Definitions

Definition 6 (Convexity) *A digital object O is convex if and only if the digital points contained in belong to O only.*

According to definition 6, the object illustrated in Fig. 1.a is convex, whereas the object illustrated in Fig. 3.a is not convex.

Definition 7 hereafter is the analog of definition 6 for parts of digital boundaries and defines convex and concave parts with respect to the clockwise orientation of the digital boundary.

Definition 7 (Convex and concave parts (Fig. 3.b and 3.c)) *Let (B_iB_j) be a part of a digital boundary B. The shortest polygonal line linking B_i and B_j located on the left (resp. right) of (B_iB_j) is called the* hull *of (B_iB_j) and is denoted by $\overline{(B_iB_j)}$ (resp. $\underline{(B_iB_j)}$). (B_iB_j) is convex (resp. concave) if there is no digital point between the polygonal line linking the points of (B_iB_j) and its hull $\overline{(B_iB_j)}$ (resp. $\underline{(B_iB_j)}$).*

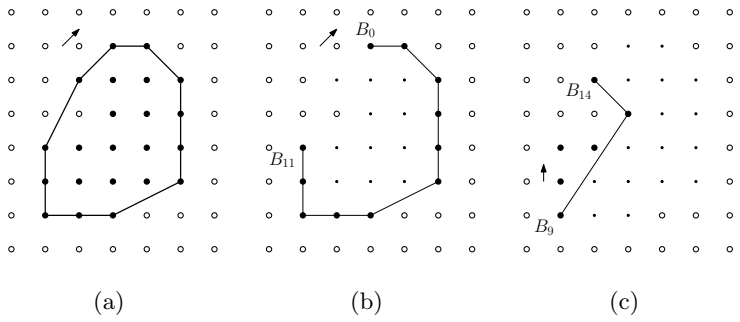


Fig. 3. In (a), the digital object depicted with black points is not convex because its convex hull contains one background point. In (b), the part (B_0B_{11}) is convex because there is no digital point between $\overline{(B_0B_{11})}$ and (B_0B_{11}). Conversely, in (c), the part (B_9B_{14}) is concave because there is no digital point between $\underline{(B_9B_5)}$ and (B_0B_{14})

Similarly to the maximal segments, we define maximal convex and concave parts:

Definition 8 (Maximal convex or concave parts) *A convex part (B_iB_j) that cannot be extended at the front (resp. at the back),* i.e. *(B_iB_{j+1}) (resp. $(B_{i-1}B_j)$) is not convex, is said maximal at the front (resp. at the back). Moreover, a maximal convex part is both maximal at the front and maximal at the back. Maximal concave parts are similarly defined.*

3.2 Main Results

A part (B_iB_j) contains a part (B_iB_l) (with $i < l < j$) that is supposed to be convex. The case where (B_iB_l) is concave is symmetric.

Proposition 1 (proved in [DRRRD03]) shows in which cases the convex part (B_iB_l) is not maximal.

Proposition 1 *Let (B_kB_l) (with $i < k < l$) be a DSS that is maximal at the back. (B_iB_{l+1}) is not convex if and only if B_{l+1} is strongly exterior to the left of (B_kB_l).*

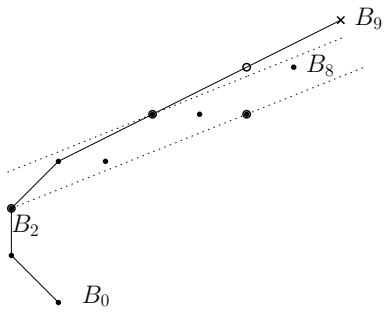


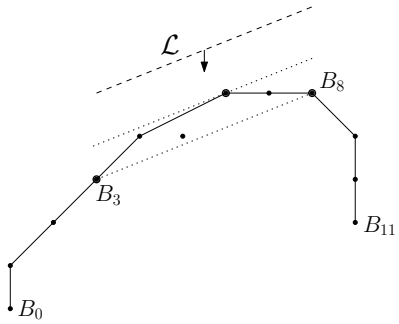
Fig. 4. (B_2B_8) is contained in a DSL of slope $\frac{2}{5}$. B_9 is strongly exterior to the left of this DSL, so (B_0B_9) is not convex

Furthermore, as shown in Lemma 1, there is a link between leaning points of maximal segments and vertices of hulls. The part (B_iB_j) is supposed to be convex. The case where (B_iB_j) is concave is symmetric.

Lemma 1 *Let (B_iB_j) contain a maximal segment (B_kB_l). The leaning points on the left of (B_kB_l) are vertices of $\overline{(B_iB_j)}$.*

Proof. Let $\frac{a}{b}$ be the slope of the strictly bounding DSL of (B_kB_l). Since (B_kB_l) is a maximal segment and (B_iB_j) is convex, all the points of (B_iB_j) that are not in (B_kB_l) are on the right of the strictly bounding DSL of (B_kB_l) (Fig. 5). Let $\mathcal{L}$ be a straight line of slope $\frac{a}{b}$ that is on the left of the strictly bounding DSL of (B_kB_l). The first points hit by $\mathcal{L}$ while $\mathcal{L}$ is moving toward (B_kB_l) are the leaning points on the left of (B_kB_l). By definition, they are vertices of $\overline{(B_iB_j)}$ too.

The number of vertices of the convex hull of a convex boundary is greater that the number of its maximal segments [dVLF07]. Thus, we cannot retrieve all the vertices of $\overline{(B_iB_j)}$ from the leaning points of the maximal segments of (B_iB_j). However, we can retrieve them in the course of the maximal segments computation, from the leaning points of segments that are not maximal, but either maximal at the front or at the back.



(a)

Fig. 5. The leaning point on the left of the maximal segment (B_3B_8) is a vertex of $\overline{(B_0B_{11})}$ because (B_0B_{11}) is convex.

Proposition 2 and Corollary 1 define the two events that provide a way of finding, from a known vertex of $\overline{(B_iB_j)}$, the next vertex of $\overline{(B_iB_j)}$. Again, a part (B_iB_j) contains a part (B_iB_l) (with $i < l < j$) that is supposed to be convex. The case where (B_iB_l) is concave is symmetric.

Proposition 2 *Let (B_kB_l) (with $i < k < l$) be a DSS that is maximal at the back. If the point B_{l+1} is exterior to the right of the strictly bounding DSS of $\overline{(B_kB_l)}$, then the last leaning point on the left of (B_kB_l) is the next vertex of $\overline{(B_iB_j)}$.*

Proof. Let us denote by $\mathcal{L}$ the straight line passing through the first and last leaning points on the left of (B_kB_l) (solid line of Fig. 6). By definition, $B_{l+1} + \vec{s}$ is located on $\mathcal{L}$ if B_{l+1} is *weakly* exterior to the right of $\mathcal{D}$ (Fig. 6) and strictly on the right of $\mathcal{L}$ if B_{l+1} is *strongly* exterior to the right of $\mathcal{D}$.

Let us denote by α the slope of $\mathcal{L}$. On the one hand, any straight line of slope greater than α that leaves the leaning points on the left of (B_kB_l) on its right side, leaves $B_{l+1} + \vec{s}$ on its right side too. Since (B_iB_j) is convex, such a line cannot contain an edge of $\overline{(B_iB_j)}$. On the other hand, any straight line that separates the leaning points on the left of (B_kB_l) from $B_{l+1} + \vec{s}$ has necessarily a slope lower than α (like the dashed line in Fig. 6).

As (B_kB_l) is maximal at the back, the first leaning point on the left of (B_kB_l) is a vertex of $\overline{(B_iB_j)}$ (Lemma 1). Thus, the last leaning point on the left of (B_kB_l) is the next vertex of $\overline{(B_iB_j)}$. □

Corollary 1 *Let (B_kB_l) (with $i < k < l$) be a DSS that is maximal at the front. If (B_kB_l) has got only one leaning point and if $(B_{k+1}B_l)$ has got strictly more than one leaning point, then the last leaning point on the left of $(B_{k+1}B_l)$ is the next vertex of $\overline{(B_iB_j)}$.*

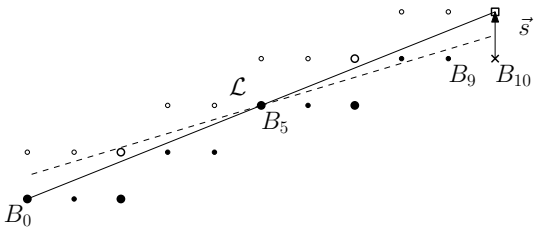
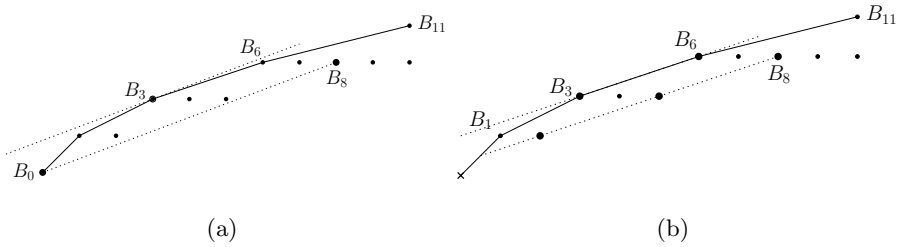


Fig. 6. B_{10} is weakly exterior to $\mathcal{D}(2,5,0)$. The first and last leaning points on the left of (B_0B_9) (B_0 and B_5) and $B_{10}+\vec{s}$ are collinear. Since B_0 is a vertex of $\overline{(B_0B_{10})}$ by hypothesis, B_5 is the next vertex of $\overline{(B_0B_{10})}$.

Fig. 7 illustrates Corollary 1.



(a) (b)

Fig. 7. B_0 is removed from the segment (B_0B_8) that is maximal at the front. Since B_3 is a vertex of $\overline{(B_0B_{11})}$ by hypothesis, B_6 is the next vertex of $\overline{(B_0B_{11})}$.

The proof of Corollary 1 is omitted because it is similar to the one of Proposition 2.

4 New and Revisited Algorithms

In this section, algorithms are derived from Proposition 1 and Proposition 2 in order to (i) decompose a digital boundary into convex and concave parts (Section 4.1), (ii) extract the hull of each convex or concave part (Section 4.2), (iii) compute polygonal representations respecting convex and concave parts (Section 4.3) and (iv) perform the preprocessing stage that optimizes the digital arc segmentation (Section 4.4).

4.1 Decomposition into Convex and Concave Parts

A simple online and linear-time algorithm to decompose a given part of boundary into convex and concave subparts is derived from Proposition 1 (Algorithm 1).

For sake of clarity, we assume in Algorithm 1 that the first retrieved subpart is convex. The core of the algorithm is the scan of a part with a window corresponding to a DSS as maximal as possible, like in the maximal segments computation [FT99,LVdV07]. If a point that is strongly exterior to the left (resp. right) of the current DSS is found (line 6), then a convex (resp. concave) part is retrieved (line 7) and a new concave (resp. convex) part is searched (lines 8-9).

Algorithm 1: Decomposition into convex and concave parts

Input: a part (B_iB_j) of a boundary B
Output: The list $\mathcal{L}$ of convex and concave parts of (B_iB_j)

```
1  L ← ∅ ;                                  /* list of convex and concave parts */
2  x ← true ;                              /* true if convex and false otherwise */
3  k' ← i ;                                 /* beginning of the current segment */
4  k ← k' ;          /* beginning of the current convex or concave part */
5  l ← i + 1 ;                                  /* index of the current point */
6  while l ≤ j do
7      l ← l + 1 ;                                /* add a point to the front */
8      if (B_k B_l) is not a DSS then
           /* side(x) returns 'left' if x is true, 'right' otherwise   */
9          if B_l is strongly exterior and on the (side(x)) of (B_k B_{l-1}) then
10             L ← L + (B_{k'} B_{l-1}) ; /* add this part to the decomposition */
11             k' ← k;
12             x ← ¬x ;          /* from convex to concave and vice versa */
13         if B_l is strongly exterior and on the (side(¬x)) of (B_k B_{l-1}) then
14             while (B_k B_l) is not a DSS do
15                 k ← k + 1 ;               /* remove a point from the back */
16 return L + (B_{k'} B_{l-1});
```

Contrary to [Fes05] and [DRDR05], the algorithm is online, because the decomposition is not derived from the slopes of the maximal segments but is given in the course of the maximal segments computation.

In [CGRT04], the algorithm of [DRRRD03] for testing the convexity is used in order to perform an online decomposition into convex and concave parts. But the decomposition is greedy and results in a set of pairwise disjoint parts of the boundary. Our decomposition is unique and results in a set of maximal convex and concave parts. Furthermore, in the algorithm of [DRRRD03], each point is processed three times at most, according to the authors, whereas in Algorithm 1, because removing a point at the back of a DSS [LVdV07] is allowed, each point is processed twice at most.

Fig. 8.b illustrates the decomposition. Notice that, as expected, the part $(B_{27}B_{38})$, which is contained both in the convex part (B_0B_{38}) and the concave one $(B_{27}B_{66})$, is a maximal segment.

In the next subsection, we go further and propose an online and linear-time algorithm that provides the hull of each convex or concave parts.

4.2 Hull of Each Convex or Concave Parts

From Proposition 2 and Corollary 1, an online and linear-time algorithm is derived to extract the hull of each convex and concave part (Algorithm 2).

Moreover, the Bezout points of the edges of the hull, that is the Bezout points of the DSS whose extremities are the vertices of the edges of the hull, are naturally extracted at the same time. Indeed, as the edges are given by the leaning points on the left or right of a DSS, their Bezout points are computed from the opposite leaning points of the DSS, thanks to $\vec{s}$ (Section 2.2).

Algorithm 2: Hull of each convex or concave part and its Bezout points

Input: a part $(B_i B_j)$ of a boundary B
Output: The list $\mathcal{L}$ of leaning and Bezout points

```
1  x ← true ;                          /* true if convex and false otherwise */
2  k ← i ;                               /* beginning of the current segment */
3  l ← i + 1 ;                                 /* index of the current point */
4  L ← B_k ;                            /* list of leaning and Bezout points */
5  while l ≤ j do
6      l ← l + 1 ;                                  /* add a point to the front */
       /* side(x) means 'left' if x is true, 'right' otherwise              */
7      if (B_k B_l) is a DSS then
8          if B_l is weakly exterior and on the (side(¬x)) of (B_k B_{l-1}) then
9              L ← L + the list of Bezout points on the (side(¬x)) of (B_k B_{l-1})
               that are between the first and last opposite leaning points;
10             L ← L + the last leaning point on the (side(x)) of (B_k B_{l-1});
11     else
12         if B_l is strongly exterior and on the (side(x)) of (B_k B_{l-1}) then
13             x ← ¬x ;              /* from convex to concave and vice versa */
14         if B_l is strongly exterior and on the (side(¬x)) of (B_k B_{l-1}) then
15             while (B_k B_l) is not a DSS do
16                 k ← k + 1 ;                  /* remove a point from the back */
17                 if (B_{k-1} B_{l-1}) has a unique leaning point on the (side(x)) and
                   (B_k B_{l-1}) has more than one leaning point on the same side
                   then
18                     L ← L + the list of Bezout points on the (side(¬x)) of
                       (B_k B_{l-1}) between the first and last opposite leaning points;
19                     L ← L + the last leaning point on the (side(x)) of (B_k B_{l-1});
20 return L;
```

Algorithm 2 has an invariant : the first leaning point on the left or the right of the current segment is always a vertex of the hull of the current part. Since the current segment is either maximal at the front or at the back or both during the maximal segments computation, the assumptions of Proposition 1, Proposition 2 and Corollary 1 are fulfilled. Proposition 2 and Corollary 1 show that the invariant is valid and guarantee that Algorithm 2 is correct.

To process strictly convex or concave parts is straightforward. However, a change of convexity brings trickier issues because two hulls enclose a part that is both convex and concave. As described in Section 4.3, the last points of the first hull and the first points of the second hull are not stored in the list in order to correctly link the two hulls.

4.3 Polygonal Representations

The output of Algorithm 2 may be modified to get different meaningful lists of points. In Fig. 8.c, the set of black points depicts the leaning points retrieved by Algorithm 2. The black polygonal line that goes through the black points is a polygonal line respecting convex and concave parts. Indeed, in the strictly convex and concave parts, the polygonal line is equal to their hull. In the parts that are both convex and concave, the first leaning point of the maximal segment of inflection is linked with the last one. For instance, in the maximal segment of inflection $(B_{27}B_{38})$, B_{29} is linked with B_{36}.

In Fig. 8.d, the set of black points depicts the leaning points retrieved by Algorithm 2 in the convex parts. The set of white points depicts the leaning points retrieved by Algorithm 2 in the concave parts, but shifted by $\vec{s}$. For instance, as the DSS from which B_{36} has been extracted is in the first octant, $B_{36} + \vec{s} = B_{36} + (0, 1)$. The black polygonal line that goes through the whole set of points faithfully represents the convex and concave parts. It is not hard to show that this polygonal line is actually the minimum-length polygon (MLP) between O and $\bar{O}$ [SCH72]. The MLP is reversible if two digitization schemes are considered at the same time: Object Boundary Quantification (OBQ) for convex parts and Background Boundary Quantification (BBQ) for concave parts. In other words, the MLP of O equals the MLP of $\bar{O}$. Thanks to these properties, the MLP is a good perimeter estimator as experimentally shown in [CK04].

4.4 Digital Arc Segmentation

The list of points extracted thanks to Algorithm 2 is useful for circular arc segmentation too. A common approach to the circular arc recognition is to search Euclidean circles separating O from $\bar{O}$. Elementary algorithms that retrieve the set of separating circles are computationally expensive. That is why an optimisation has been proposed in [CGRT04]. For each convex or concave part, the set of points that have to be enclosed by the separating circles can be reduced to the hull of the part. Moreover, the set of points that have not to be enclosed by the separating circles can be reduced to some of the Bezout points of the edges of the hull: those that are located near the bisector of each edge.

In Fig 8.e and 8.f, the black polygonal line separates O from $\bar{O}$. The set of black points depicts the vertices of the MLP. In Fig 8.e, the set of white points depicts the Bezout points of the edges of the MLP. In Fig 8.f, only those that are located near the bisector of each edge are depicted. An approach to the circular arc segmentation is to iteratively search Euclidean circles separating the points of the MLP from some of its Bezout points (Fig 8.f).

In [CGRT04], the extraction of these points consists of three steps. First, the procedure of [DRRRD03] is used to decompose the digital curve into convex and concave parts. Then, each part is decomposed into DSS [DRR95]. Finally, the hull and Bezout points of each segment are computed with the extended Euclid's algorithm. Algorithm 2 provides in one scan the hull and Bezout points of each convex or concave part. Taking only the Bezout points located near the bisector of the two leaning points instead of all (lines 7 and 16 of Algorithm 2) is sufficient to simply perform in one scan what is done in three steps in [CGRT04].

5 Conclusion

Algorithm 1 and Algorithm 2 are both similar to the algorithm that computes the set of maximal segments [LVdV07]. As a consequence we can merge these algorithms to get in one scan: (i) the set of maximal segments (Fig. 8.a) (ii) the convex and concave parts (Fig. 8.b) (iii) the hull of each convex or concave part and their Bezout points (Fig. 8.e). These data are useful for polygonal representation (Fig. 8.c and 8.d) as well as for decomposition into circular arcs (Fig. 8.f).

The algorithms presented in this paper are considerably neater than previous ones [DRRRD03,DRDR05,CGRT04] because only one scan by a window corresponding to a segment maximal either at the front or at the back is performed. Each point is processed in a constant time twice at most and thus, the whole algorithm is of order $\mathcal{O}(n)$ for a part of a digital boundary having n points.

References

[CGRT04] D. Coeurjolly, Y. Gérard, J-P. Reveillès, and L. Tougne. An Elementary Algorithm for Digital Arc Segmentation. *Discrete Applied Mathematics*, 139(1-3):31–50, 2004.

[CK04] D. Coeurjolly and R. Klette. A Comparative Evaluation of Length Estimators of Digital Curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:252–257, 2004.

[CMT01] D. Coeurjolly, S. Miguet, and L. Tougne. Discrete Curvature Based on Osculating Circle Estimation. In *4th International Workshop on Visual Form*, 2001.

[DRDR05] H. Dorksen-Reiter and I. Debled-Rennesson. Convex and Concave Parts of Digital Curves. *Geometric Properties from Incomplete Data*, 31:145–159, 2005.

[DRDR06] H. Dorksen-Reiter and I. Debled-Rennesson. A linear Algorithm for Polygonal Representations of Digital Sets. In *International Workshop in Combinatorial Image Analysis*, pages 307–319, 2006.

[DRR95] I. Debled-Rennesson and J-P. Reveillès. A linear algorithm for segmentation of digital curves. *International Journal of Pattern Recognition and Artificial Intelligence*, 9:635–662, 1995.

[DRRRD03] I. Debled-Rennesson, J-L. Rémy, and J. Rouyer-Degli. Detection of the Discrete Convexity of Polyominoes. *Discrete Applied Mathematics*, 125:115–133, 2003.

[dVLF07] F. de Vieilleville, J.-O. Lachaud, and F. Feschet. Maximal Digital Straight Segments and Convergence of Discrete Geometric Estimators. *Journal of Mathematical Image and Vision*, 27(2):471–502, 2007.

[Eck01] U. Eckhardt. Digital Lines and Digital Convexity. In *Digital and Image Geometry*, pages 209–227, 2001.

[EDR04] U. Eckhardt and H. Dorksen-Reiter. Polygonal Representations of Digital Sets. *Algorithmica*, 38(1):5–23, 2004.

[Fes05] F. Feschet. Canonical Representations of Discrete Curves. *Pattern Analysis and Applications*, 8:84–94, 2005.

[FT99] F. Feschet and L. Tougne. Optimal Time Computation of the Tangent of a Discrete Curve: Application to the Curvature. In *Discrete Geometry in Computer Imagery*, pages 31–40, 1999.

[FT05] F. Feschet and L. Tougne. On the Min DSS Problem of Closed Discrete Curves. *Discrete Applied Mathematics*, 151:138–153, 2005.

[LVdV07] J.-O. Lachaud, A. Vialard, and F. de Vieilleville. Fast, Accurate and Convergent Tangent Estimation on Digital Contours. *Image and Vision Computing*, 25:1572–1587, 2007.

[Rev91] J-P Reveillès. *Géométrie Discrète, calculs en nombres entiers et algorithmique.* thèse d'etat, Université Louis Pasteur, 1991.

[SCH72] J. Sklansky, R. L. Chazin, and B. J. Hansen. Minimum-perimeter Polygons of Digitized Silhouettes. *IEEE Transactions on Computers*, 21(3):260–268, 1972.

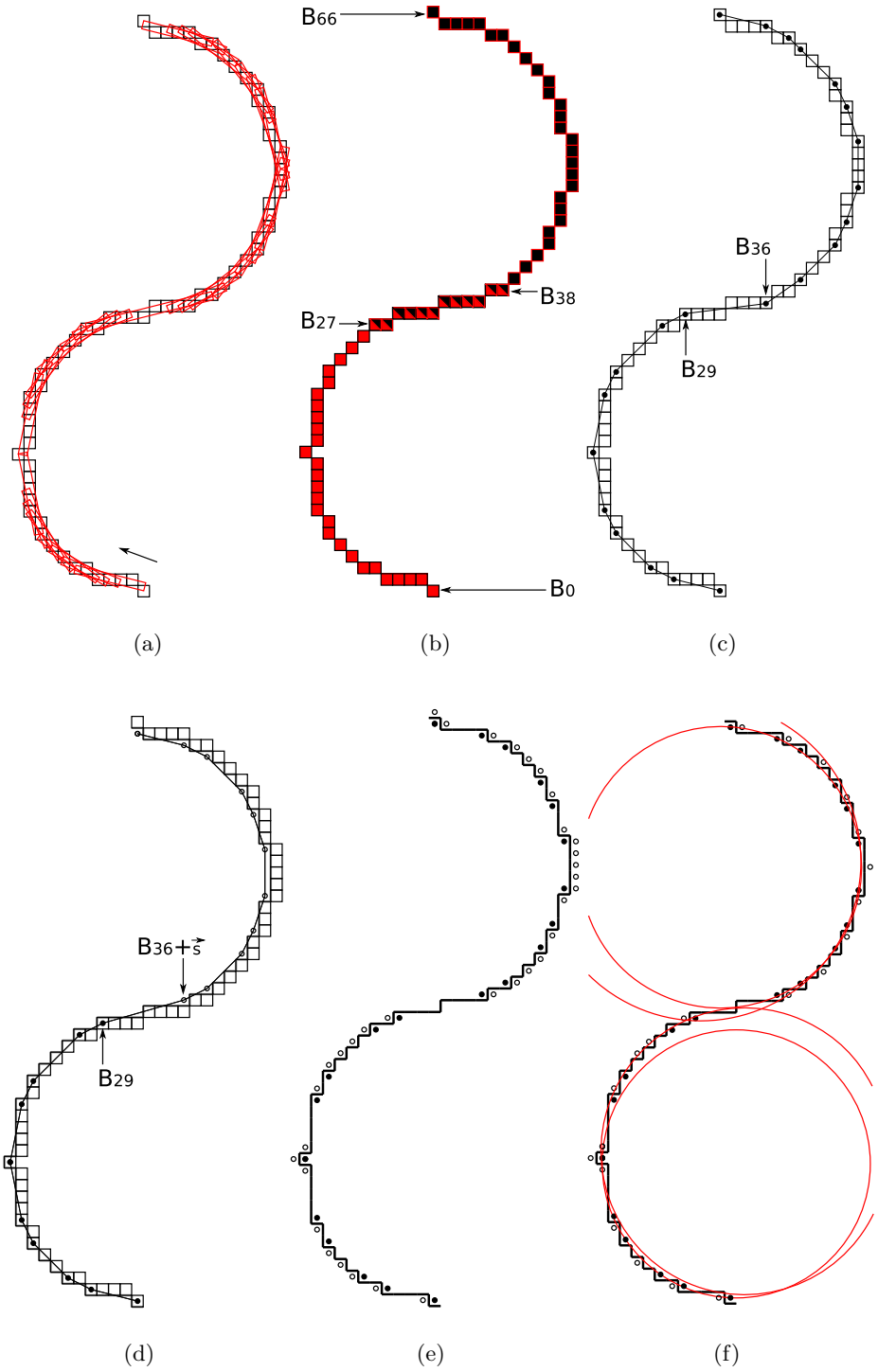


Fig. 8. (a) Maximal segments, (b) Convex and concave parts, (c) Polygonal representation respecting convex and concave parts, (d) Minimum-Length Polygon, (e) Minimum-Length Polygon with its associated Bezout points, (e) List of sufficient points for the digital arc segmentation