Discrete Applied Mathematics 159 (2011) 2229–2250

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Two linear-time algorithms for computing the minimum length polygon of a digital contour☆

J.-O. Lachaud [a], X. Provençal [a,b,*]

[a] LAMA, UMR CNRS 5127, Université de Savoie, F-73376 Le Bourget du Lac, France

[b] LIRMM, UMR CNRS 5506, Université Montpellier II, 161 rue Ada, F-34000 Montpellier, France

ARTICLE INFO

Article history:
Received 4 May 2010
Received in revised form 14 July 2011
Accepted 1 August 2011
Available online 8 September 2011

Keywords:
Discrete geometry
Shape analysis
Minimum length polygon
Multigrid convergent length estimator
Combinatorics on words

ABSTRACT

The Minimum Length Polygon (MLP) is an interesting first order approximation of a digital contour. For instance, the convexity of the MLP is characteristic of the digital convexity of the shape, its perimeter is a good estimate of the perimeter of the digitized shape. We present here two novel equivalent definitions of MLP, one arithmetic, one combinatorial, and both definitions lead to two different linear time algorithms to compute them. This paper extends the work presented in Provençal and Lachaud (2009) [26], by detailing the algorithms and providing full proofs. It includes also a comparative experimental evaluation of both algorithms showing that the combinatorial algorithm is about 5 times faster than the other. We also checked the multigrid convergence of the length estimator based on the MLP.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

The minimum length polygon (MLP) or minimum perimeter polygon was proposed long ago for approaching the geometry of a digital contour [25,30]. One of its definitions is to be the polygon of minimum perimeter which stays in the band of 1 pixel-wide centered on the digital contour. It has many interesting properties such as: (i) it is reversible [25]; (ii) it is characteristic of the convexity of the digitized shape and it minimizes the number of inflexion points to represent the contour [30,15]; (iii) it is a good digital length estimator [19,6] and is proven to be multigrid convergent in $O(h)$ for digitization of convex shapes, where h is the grid step (reported in [18,31,32]); (iv) it is also a good tangent estimator; (v) it is the relative convex hull of the digital contour with respect to the outer pixels [30,33] and is therefore exactly the convex hull when the contour is digitally convex.

Several algorithms for computing the MLP have been published. We have already presented the variational definition of the MLP (length minimizer). It can thus be solved by a nonlinear programming method. The initial computation method of [25] was indeed an interactive Newton–Raphson algorithm. Computational complexity is clearly not linear and the solution is not exact. We have also mentioned its set theoretic definition (intersection of relative convex sets). However, except for digital convex shapes, this definition does not lead to a specific algorithm. The MLP may also be seen as a solution to a shortest path query in some well chosen polygon. An adaptation of [14] to digital contour could be implemented in time linear with the size of the contour. It should however be noted that data structures and algorithms involved are complex and difficult to implement. Klette et al. [17] (see also [19,18]) have also proposed an arithmetic algorithm to compute it, but

☆ This paper is an extended version of Provençal and Lachaud (2009) [26], published in Proc. DGCI'2009, LNCS 5810, Springer. This work was partially supported by FQRNT (Québec) and ANR project FOGRIMMI (ANR-06-MDCA-008-06).

* Corresponding author. Fax: +33 4 67 41 85 00.
E-mail addresses: jacques-olivier.lachaud@univ-savoie.fr (J.-O. Lachaud), xavier.provencal@univ-savoie.fr (X. Provençal).

0166-218X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2011.08.002

as it is presented, it does not seem to compute the MLP in all cases. As reported in [9], its edges seem restricted to digital straight segments such that the continued fraction of their slope has a complexity no greater than two.

The MLP is in some sense characteristic of a digital contour. One may expect to find strong related arithmetic and combinatorial properties. This is precisely the purpose of this paper. Furthermore, we show that each of these definitions induces an optimal time integer-only algorithm for computing it. The combinatorial algorithm is particularly simple and elegant, while the arithmetic definition is essential for proving it defines the MLP. These two new definitions give a better understanding of what is the MLP in the digital world. Although other linear-time algorithms exist, the two proposed algorithms are simpler than existing ones. They are thus easier to implement and their constants are better.

The paper is organized as follows. First Section 2 recalls standard definitions. Section 3 gives formally the above-mentioned alternative definitions of the MLP. Section 4 presents how to split uniquely a digital contour into convex, concave and inflexion zones, the arithmetic definition of MLP follows then naturally. Section 5 is devoted to the combinatorial version of MLP. After establishing its equivalence with the arithmetic MLP, we show that our algorithm constructs it in linear time. Section 6 illustrates our results and concludes.

This paper is an extended version of [26]. We provide here full proofs and further examples. We also note that an algorithm for computing the MLP has just been proposed independently by Roussillon et al. (to appear in [28]): it is extremely similar in spirit to our arithmetic algorithm since its computation relies also on maximal segment recognition. However our combinatorial MLP should still be much faster in practice since it does not compute the geometry of segments along the shape.

2. Preliminaries

This section presents the standard definitions that we will used throughout the paper, in order to avoid any ambiguity.

2.1. Polyomino, digital contour, inner and outer polygon

Given some set X in the plane, its *topological interior* will be denoted by X° while its *topological boundary* will be denoted by ∂X.

A *digital square* is a unit closed axis-aligned square in the plane whose center has integer coordinates. A *polyomino* is a set of digital squares in the plane such that its topological boundary is a Jordan curve. It is thus bounded. It is convenient to represent a polyomino as a subset of the digital plane $\mathbb{Z}^2$, which codes the integer coordinates of the centers of its squares, instead of representing it as a subset of the Euclidean plane $\mathbb{R}^2$. When seeing a polyomino as a subset of $\mathbb{R}^2$, we will say *the body of the polyomino*. For instance, the Gauss digitization of a convex subset of the plane is a polyomino iff it is 4-connected. A subset of $\mathbb{Z}^2$, or *digital shape*, is a polyomino iff it is 4-connected and its complement is 4-connected.

In the following, we call *digital contour* the boundary of any polyomino, represented as a sequence of horizontal and vertical steps in the half-integer plane $\left(\mathbb{Z}+\frac{1}{2}\right)\times\left(\mathbb{Z}+\frac{1}{2}\right)$. One can use for instance a Freeman chain to code it as a word over the alphabet $\{0, 1, 2, 3\}$. These words are usually called *contour words*. Again, the *body* of a digital contour is its embedding in $\mathbb{R}^2$ as a polygonal curve. Now, since the body of a digital contour is a Jordan curve, it has one well-defined inner component in $\mathbb{R}^2$, whose closure is exactly the polyomino whose boundary is the digital contour. There is thus a one-to-one map from digital contours to polyominoes, denoted by I.

Let Sq be the digital square centered at $(0, 0)$ and let $\oplus$ denotes the Minkowski sum of two sets.

We only deal in this paper with *simple* digital contours (or grid continua in the terminology of [32]). A digital contour C is *simple* if and only if: (i) any digital point of a digital contour C has exactly in its 4-neighborhood two other digital points of C, (ii) the one pixel-wide band $C \oplus Sq$ is an annulus whose topological boundary is composed of two simple closed polygonal lines.

Each of these lines induces a finite simple polygon by Jordan's theorem. The one included in the body of $\mathrm{I}(C)$ is called the *inner polygon* of C and is denoted by $\mathrm{L}_1(C)$. The other one is the *outer polygon* of C and is denoted by $\mathrm{L}_2(C)$. We have thus by definition that $C \oplus Sq = \mathrm{L}_2(C) \setminus \mathrm{L}_1(C)^\circ$. It is easy to check that all digital points on $\partial\mathrm{L}_1(C)$ are in the polyomino $\mathrm{I}(C)$ while all digital points on $\partial\mathrm{L}_2(C)$ are not in the polyomino $\mathrm{I}(C)$. These notions are illustrated on Fig. 1.

2.2. Maximal segments; tangential cover; turns

A *standard digital straight line (DSL)* is some set $\{(x, y) \in \mathbb{Z}^2,\ \mu \le ax-by < \mu+|a|+|b|\}$, where (a, b, μ) are also integers and $\gcd(a, b) = 1$. It is well known that a DSL is a 4-connected simple path in the digital plane, which is the digitization of a Euclidean straight line of slope $\frac{a}{b}$ and shift to origin $-\frac{\mu}{b}$ [27,7]. A *digital straight segment (DSS)* is a 4-connected piece of DSL. Given a digital contour C, a *maximal segment* M is a subset of C that is a DSS and which is no more a DSS when adding any other point of $C \setminus M$.

We recall that the *tangential cover* of a digital contour is the ordered sequence of its maximal segments [12]. In the following, the tangential cover is denoted by $(M_l)_{l=0\cdots m-1}$, where M_l is the l-th maximal segment of the contour. Let us denote by θ_l the slope direction (angle wrt x-axis) of M_l. All indices are taken modulo the number m of maximal segments. Since the directions of two consecutive maximal segments can differ of no greater than π, their variation of direction can

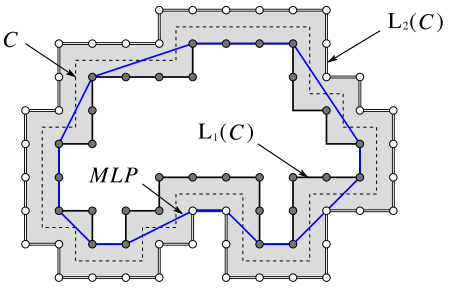


Fig. 1. A digital contour C with its inner polygon $\mathrm{L}_1(C)$, its outer polygon $\mathrm{L}_2(C)$ and its MLP.

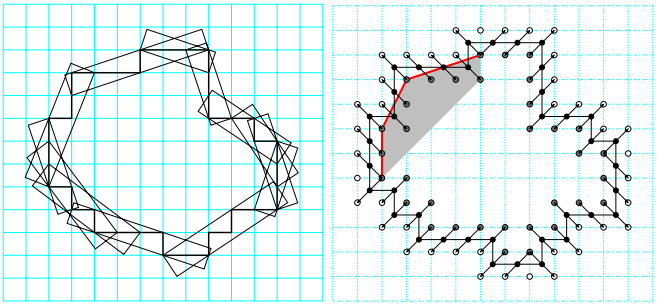


Fig. 2. Left: digital contour, tangential cover. Right: inside and outside pixels and, in red, edges of AMLP(C) in a convex part of C. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

always be casted in $]-\pi,\pi[$ without ambiguity. The angle variation $(\theta_l-\theta_{l+1}) \mod [-\pi,\pi[$ is denoted by $\Delta(\theta_l,\theta_{l+1})$. For clarity, we will also write $\theta_l > \theta_{l+1}$ when $\Delta(\theta_l,\theta_{l+1}) > 0$. We always consider the digital contour to turn clockwise around the polyomino. A couple of consecutive maximal segments (M_l, M_{l+1}) is thus said to be a $\wedge$-*turn* (resp. $\vee$-*turn*) when $\Delta(\theta_l,\theta_{l+1})$ is negative (resp. positive). The symbol $\wedge$ stands for "convex" while the symbol $\vee$ stands for "concave".

Since a maximal segment is contained in a digital straight line, it is formed of exactly two kinds of steps, with Freeman codes c and $(c+1) \mod 4$. This coding defines the *quadrant* of the maximal segment. Its *quadrant vector* is then the diagonal vector that is the sum of the two unit steps coded by the Freeman codes of the quadrant, rotated by $+\frac{\pi}{2}$.

We eventually associate pixels to contour points (C_i) as follows:

- the *inside pixel* $\mathrm{in}(C_i)$ of C_i is the pixel $C_i - \frac{\vec{v}}{2}$, where $\vec{v}$ is the quadrant vector of any maximal segment containing it (or the last maximal segment strictly containing it at a quadrant change).
- the *outside pixel* $\mathrm{out}(C_i)$ of C_i is the pixel $C_i + \frac{\vec{v}}{2}$, where $\vec{v}$ is the quadrant vector of any maximal segment containing it (or the last maximal segment strictly containing it at a quadrant change).

Fig. 2 illustrates these definitions. It is clear that inside pixels belong to $\partial\mathrm{L}_1(C)$ and outside pixels to $\partial\mathrm{L}_2(C)$.

3. Existing definitions of the MLP

We recall and give formally several definitions for the MLP of a digital contour. The first one relates it to the standard convex hull for convex digital contours. The second one extends naturally this definition to arbitrary simple contours as the intersection of specific subsets of the plane. This definition of MLP is the most convenient in our case for proving our results. The third one is the classical definition of MLP as the solution to a variational problem.

3.1. Variational definition

Following the works of Sloboda et al. [31,34,33] (or see [17,18]), we define the minimum length polygon (MLP) of C as the shortest Jordan curve whose digitization is (very close to) the polyomino of C. More precisely, letting $\mathcal{A}$ be the family of simply connected compact sets of $\mathbb{R}^2$, we define the following.

Definition 1. The *minimum perimeter polygon* of two polygons V, U with $V \subset U^\circ \subset \mathbb{R}^2$ is a subset P of $\mathbb{R}^2$ such that

$$P = \operatorname*{argmin}_{A\in\mathcal{A}, V\subseteq A, \partial A\subset U\setminus V^\circ} \mathrm{Per}(A), \tag{1}$$

where $\mathrm{Per}(A)$ stands for the perimeter of A, more precisely the 1-dimensional Hausdorff measure of the boundary of A.

Definition 2. The *minimum length polygon (MLP)* of a digital contour C is the minimum perimeter polygon of $\mathrm{L}_1(C)$, $\mathrm{L}_2(C)$.

3.2. Set-theoretic definition

The relative convex hull leads to a nice and simple set-theoretic definition of the MLP. This definition is very general since it is related to sets in n-dimensional Euclidean spaces. It is a rather natural extension of convex hull. In the following, the notation $\overline{xy}$ stands for the straight line segment joining x and y, i.e. their convex hull.

Definition 3 (*[33]*)**.** Let $U \subseteq \mathbb{R}^n$ be an arbitrary set. A set $C \subseteq U$ is said to be *U-convex* iff for every $x, y \in C$ with $\overline{xy} \subseteq U$ it holds that $\overline{xy} \subseteq C$.

Let $V \subseteq U \subseteq \mathbb{R}^n$ be given. The intersection of all U-convex sets containing V will be termed *convex hull of V relative to U*, or more shortly *U-convex hull of V*, and denoted by $\mathrm{Conv}_U(V)$.

We use this definition in the 2-dimensional case.

Definition 4. The *set-theoretic MLP* of a digital contour C is the convex hull of $\mathrm{L}_1(C)$ relative to $\mathrm{L}_2(C)$.

3.3. Equivalence; convex case

The two previous definitions (MLP and set-theoretic MLP) are equivalent due to the following theorem:

Theorem 5 (*Theorem 3, [31,33]*)**.** *Eq. (1) has a unique solution, which is*

1. *a polygonal Jordan curve whose convex vertices (resp. concave) belong to the vertices of the inner polygon (resp. the vertices of the outer polygon),*
2. *the convex hull of V relative to U.*

Since for a 4-connected convex digital set A, the convex hull of A does not contain any other integer points (e.g. see [4]), it is clear that the convex hull of A is a $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$. It is also clear that it is included in any other $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$. The convex hull of A is then the set-theoretic MLP of A, and is therefore its MLP according to the previous theorem.

We also mention that the perimeter of the MLP is a good discrete perimeter estimator [34]. This is proved with standard results related to convex geometry [29]. The precision of the estimation is no greater than $8h$ if the digitization step is h. The MLP provides thus a multigrid convergent perimeter estimator with convergence speed $O(h)$ for convex shapes or for shapes with a finite number of inflexion points.

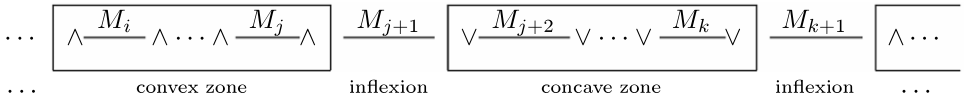
4. Arithmetic MLP

4.1. Decomposition into convex/concave/inflexion zones

We have the following theorem from [10], which relates convexity to maximal segment directions. It also induces a linear time algorithm to check convexity.

Theorem 6 (*Adapted from [10]*)**.** *A digital contour is digitally convex iff every couple of consecutive maximal segments of its tangential cover is made of $\wedge$-turns.*

For a given DSS M, its first and last upper leaning points are respectively denoted by $\mathrm{U}_f(M)$ and $\mathrm{U}_l(M)$, while its first and last lower leaning points are respectively denoted by $\mathrm{L}_f(M)$ and $\mathrm{L}_l(M)$. In the same paper, it is proven that the point $\mathrm{U}_l(M_l)$ is no further than $\mathrm{U}_f(M_{l+1})$ in the case of a convex contour. A symmetrical property holds naturally for lower leaning points in the case of a concave contour. These two properties are necessary for the consistency of points (1) and (2) of Definition 7. We note also that in any DSS M, the first upper leaning point $\mathrm{U}_f(M)$ is no further than the last lower leaning point $\mathrm{L}_l(M)$: this is due to the fact that leaning points along a DSS alternate between upper and lower position. This is used for the consistency of points (3) and (4) of Definition 7. We may now consider the succession of turns along a digital contour to cut it into parts.



Definition 7. A digital contour C is uniquely split by its tangential cover into a sequence of closed connected sets with a single point overlap as follows:

1. A *convex zone* or *$(\wedge, \wedge)$-zone* is defined by an inextensible sequence of consecutive $\wedge$-turns from (M_{l_1}, M_{l_1+1}) to (M_{l_2-1}, M_{l_2}). If $l_1 \neq l_2$, it starts at $\mathrm{U}_l(M_{l_1})$ and ends at $\mathrm{U}_f(M_{l_2})$, otherwise the digital contour is convex and constitutes a single convex zone.

2. A *concave zone* or $(\vee, \vee)$-*zone* is defined by an inextensible sequence of consecutive $\vee$-turns from $(M_{l'_1}, M_{l'_1+1})$ to $(M_{l'_2-1}, M_{l'_2})$. It starts at $\mathrm{L}_l(M_{l'_1})$ and ends at $\mathrm{L}_f(M_{l'_2})$.
3. A *convex inflexion zone* or $(\wedge, \vee)$-*zone* is defined by a $\wedge$-turn followed by a $\vee$-turn around M_i. It starts at $\mathrm{U}_f(M_i)$ and ends at $\mathrm{L}_l(M_i)$.
4. A *concave inflexion zone* or $(\vee, \wedge)$-*zone* is defined by a $\vee$-turn followed by a $\wedge$-turn around $M_{i'}$. It starts at $\mathrm{L}_f(M_{i'})$ and ends at $\mathrm{U}_l(M_{i'})$.

Note that a convex or concave zone may be reduced to a single turn between two successive inflexions. In this case, the zone may or may not be a single contour point (see Fig. 4).

4.2. Definition of the arithmetic MLP of C

The following lemma expresses the fact that a convex zone is naturally decomposed by the consecutive quadrants of its maximal segments. We recall that a polyomino is h-convex when each of its rows is connected, v-convex when each of its columns is connected and hv-convex when it is h-convex and v-convex.

Lemma 8. *Any convex zone has a unique decomposition into a factor of* $(1Q_{0<1}0Q_{3<0}3Q_{2<3}2Q_{1<2})^*$, *where* $Q_{a<b}$ *is written over the two letters* $\{a, b\}$, *begins by b and ends by b.*

Proof. A word that is not such a factor contains necessarily either some 30^k1, 23^k0, 12^k3, 01^k2 (trivial concavity), or some 02, 20, 13, 31 (back and forth), or some 103, 210, 321, 032 (one pixel wide quadrant change). The first case may not happen in a clockwise contour of a hv-convex polyomino, and therefore it may not happen in a convex zone (discrete convexity implies hv-convexity). The second case may not happen on the boundary of any polyomino. The third case may not happen on a simple digital contour. □

The words $Q_{a<b}$ are called the *quadrant words* of the convex zone. A symmetric decomposition into a factor of $(0Q_{1<0}1Q_{2<1}2Q_{3<2}3Q_{0<3})^*$ holds for the concave zones.

Definition 9. Assume $C_{i,j}$ is a connected part of a contour, with only two kinds of steps. The *left envelope* of $C_{i,j}$ is the sequence of edges of the convex hull of the inside pixels of $C_{i,j}$, such that the first vertex is the inside pixel of C_i, the last vertex is the inside pixel of C_j and the edges turn clockwise around the hull. The *right envelope* of $C_{i,j}$ is defined symmetrically by replacing everywhere inside pixel by outside pixel and clockwise by counterclockwise.

We may now define a linear analog to a digital contour which is the arithmetic MLP. We refer the reader to Fig. 3 for an illustration of this definition, see also Fig. 4 for a more detailed description of the construction of the AMLP around inflexion zones.

Definition 10. The *arithmetic* MLP (or AMLP) of a digital contour C is the polygon AMLP(C) defined by zones $C_{i,j}$ in C, according to its type:

Zone type of $C_{i,j}$	Associated part of AMLP(C)
$(\wedge, \wedge)$-zone	Union of the left envelope of each quadrant word of $C_{i,j}$
$(\vee, \vee)$-zone	Union of the right envelope of each quadrant word of $C_{i,j}$
$(\wedge, \vee)$-zone	Segment joining the inside pixel of C_i to the outside pixel of C_j
$(\vee, \wedge)$-zone	Segment joining the outside pixel of C_i to the inside pixel of C_j

4.3. The arithmetic MLP is a polygon with boundary in the band $\mathrm{L}_2(C) \setminus \mathrm{L}_1(C)^\circ$

We shall prove in this section that the arithmetic MLP of a digital contour is a polygon which separates inner pixels from outer pixels.

Lemma 11. AMLP(C) *is a closed polygonal line with vertices in* $\mathbb{Z}^2$.

Proof. We look first at a convex zone of C. According to Definition 10, the arithmetic MLP is constructed by parts for each quadrant word. The definition of envelope (Definition 9) guarantees that each part is a polygonal line. The fact that one letter is removed between two quadrant words ensures that the inside pixel of the last point of the first quadrant word is the same as the inside pixel of the first point of the second quadrant word. Therefore, each part is correctly connected to the next one and AMLP(C) is a polygonal line in a convex zone of C. The same argument holds for a concave zone. Lastly, AMLP(C) is by Definition 10 a segment in an inflexion zone.

We have just proved that AMLP(C) is a polygonal line in each zone of C. Now, any two consecutive polygonal lines share the same extremity. For instance, a proper $(\wedge, \wedge)$-zone of C, terminated at $\mathrm{in}(\mathrm{U}_f(M_l))$ may only be followed by a

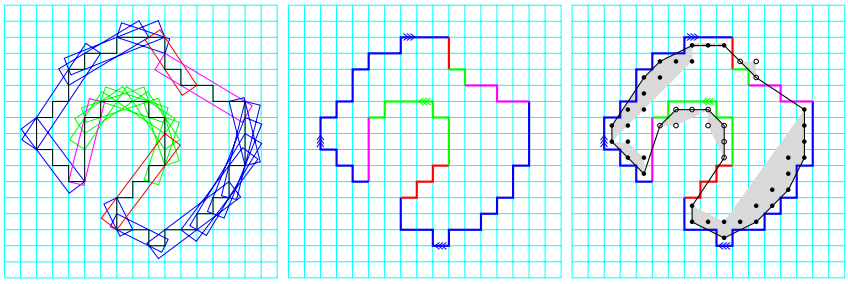


Fig. 3. Illustration of the construction of the AMLP from a digital contour. *Left*: the input digital contour is the word 1101011010010003303003003333233232322321221101010111212222323333212 12. The maximal segments define two $(\wedge, \wedge)$-zones (in blue), two $(\vee, \vee)$-zones (in green), two $(\wedge, \vee)$-zones (in red) and two $(\vee, \wedge)$-zones (in magenta). *Center*: Each convex zone is decomposed into quadrant words. For instance, the leftmost convex zone is decomposed as $21212 \cdot \underline{1} \cdot 10101101001 \cdot \underline{0} \cdot 00$, where the isolated letters $\underline{1}$ and $\underline{0}$ correspond to quadrant changes. *Right*: the AMLP is computed by zones either by convex hull computation in convex or concave zones, or just by connecting endpoints in inflexion zones. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

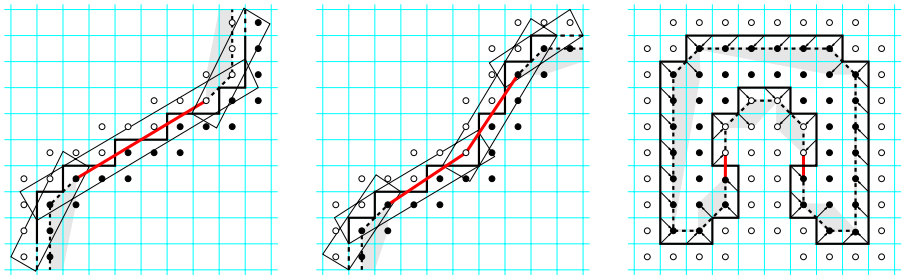


Fig. 4. Examples of AMLP: *Left*. the left envelope of a $(\wedge, \wedge)$-zone, the right envelope of a $(\vee, \vee)$-zone both joined by the segment associated to the $(\wedge, \vee)$ convex inflexion zone. *Center*. A $(\vee, \vee)$-zone is reduced to a single point bordered by two inflexion zones. *Right*. An example of AMLP with inside (resp. outside) pixels of each quadrant word.

$(\wedge, \vee)$-zone, beginning also at $\mathrm{in}(\mathrm{U}_f(M_l))$. The digital contour point is clearly the same for both, and so is its inside pixel. Indeed, since $\mathrm{U}_f(M_l)$ is the first upper leaning point of a maximal segment, the next maximal segment M_{l+1} cannot contain it. Therefore it is M_l that defines the quadrant vector for this point and thus the position of the inside pixel. Other cases are treated identically. AMLP(C) is thus a closed polygonal line whose vertices are by construction in $\mathbb{Z}^2$. □

We recall that the set $C \oplus Sq$ is also the one pixel wide band $\mathrm{L}_2(C) \setminus \mathrm{L}_1(C)^\circ$.

Lemma 12. *In a $(\wedge, \wedge)$-zone $C_{i,j}$ of C, the corresponding edges of* AMLP(C) *form a simple polygonal line included in* $\mathrm{L}_2(C)^\circ \setminus \mathrm{L}_1(C)^\circ$, *and in* $C_{i,j} \oplus Sq$.

Proof. The word $C_{i,j}$ is composed of quadrant words (Lemma 8). Let $Q_{a<b}$ be one of them. Without loss of generality, assume $a = 0$ and $b = 1$. We claim that its left envelope $E_{0<1}$ is in $Q_{0<1} \oplus Sq$.

We further denote by $\overrightarrow{v}$ the quadrant vector of a word in $\{0, 1\}^*$. Since $C_{i,j}$ is a convex zone, the sequence of maximal segments of $Q_{0<1}$ has only $\wedge$-turns. Theorem 8.1 in [10] shows this contour is digitally convex, and there is then no integer point between the upper convex hull of $Q_{0<1}$ and $Q_{0<1}$ itself. Now the inside pixels $\mathrm{in}(Q_{0<1})$ of $Q_{0<1}$ constitutes the same contour as $Q_{0<1}$ but for a translation by $-\frac{\overrightarrow{v}}{2}$. Furthermore, the left envelope $E_{0<1}$ is also the translation by $-\frac{\overrightarrow{v}}{2}$ of the upper convex hull of $Q_{0<1}$. Hence there is no integer point between $E_{0<1}$ and $\mathrm{in}(Q_{0<1})$.

We denote by P the polygon formed by the union of the two polygonal curves $E_{0<1}$ and $\mathrm{in}(Q_{0<1})$. Its interior does not contain any point of $\mathbb{Z}^2$. The only integer points that are touched by P are in $\mathrm{I}(C)$. First, P has an empty intersection with $\mathrm{L}_1(C)^\circ$. Indeed, assume there is some $x \in P \cap \mathrm{L}_1(C)^\circ$, then x would belong to an open unit square delimited by four adjacent integer points of $\mathrm{L}_1(C)$. This is not possible since the points of $\mathrm{L}_1(C)$ that may touch P form an oriented path in the quadrant $0 < 1$. A square would require a step in another direction than 0 or 1.

Furthermore, the boundary of P does not cross any unit segment between two adjacent points of $\mathbb{Z}^2 \setminus \mathrm{I}(C)$ (otherwise we could build a smaller convex set which does not cross it). Since the vertices of $\mathrm{L}_2(C)$ belong to this set of points, we conclude that P cannot intersect $\partial\mathrm{L}_2(C)$. Jordan's theorem applied to $\partial\mathrm{L}_2(C)$ induces that $P \subset \mathrm{L}_2(C)^\circ$.

We have thus that $E_{0<1} \subset P \subset \mathrm{L}_2(C)^\circ \setminus \mathrm{L}_1(C)^\circ$. It thus shows $E_{0<1} \subset C \oplus Sq$. All points defining the hull are included in $Q_{0<1} \oplus Sq$. Only an edge of P may go outside $C_{i,j} \oplus Sq$. We already know that such edges may not cross $\partial\mathrm{L}_2(C)$ or $\partial\mathrm{L}_1(C)$. The only ways out are the two unit segments $(C_{i-1} \oplus Sq) \cap (C_i \oplus Sq)$ and $(C_j \oplus Sq) \cap (C_{j+1} \oplus Sq)$. It is clear that would an edge of P go outside through these segments, then it exit through one before entering through the other. So P does a loop around $\mathrm{L}_1(C)$ within $C \oplus Sq$, and has slopes in all quadrants. This is impossible since by construction all edges of P starting from the bottom left point are on both sides with slopes in the first quadrant (along $E_{0<1}$ on one side and $\mathrm{in}(Q_{0<1})$ on the other side).

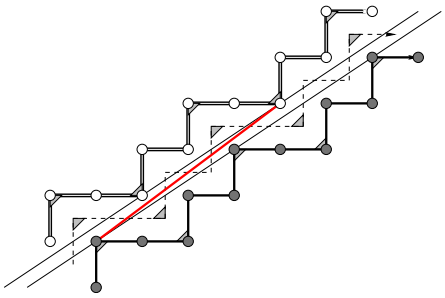


Fig. 5. Geometry of a DSS of slope $\frac{2}{3}$ along a digital contour. The digital contour is drawn in between the corresponding inside and outside pixels. They clearly draw the same contour up to a translation. Upper and lower leaning points are denoted by gray right triangles. The thick red line connects the first upper leaning point on the inside contour to the last lower leaning point on the outside contour. The two straight lines define a straight band which separates inside from outside pixels. The thick red line is in this band, thus in $\mathrm{L}_2(C) \setminus \mathrm{L}_1(C)^\circ$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Gathering every left envelope of $C_{i,j}$ we get a polygonal line from $\mathrm{in}(C_i)$ to $\mathrm{in}(C_j)$, which is included in $C_{i,j} \oplus Sq$. This polygonal line is simple since it is simple in each part and since each connection made at quadrant change is a proper vertex. □

The following lemma is proven similarly to the previous one.

Lemma 13. *In a $(\vee, \vee)$-zone $C_{i',j'}$ of C, the corresponding edges of* AMLP(C) *form a simple polygonal line included in* $\mathrm{L}_2(C) \setminus \mathrm{L}_1(C)$*, and in $C_{i',j'} \oplus Sq$.*

Lemma 14. *In a $(\wedge, \vee)$ or $(\vee, \wedge)$-zone $C_{i'',j''}$ of C, the corresponding edges of* AMLP(C) *form a single straight segment included in* $\mathrm{L}_2(C) \setminus \mathrm{L}_1(C)^\circ$*, and in $C_{i'',j''} \oplus Sq$.*

Proof. We refer the reader to Fig. 5. The AMLP(C) is in this zone the thick red segment squeezed in the band that separates inside from outside pixels. It is indeed easy to see that, if (a, b, μ) are the characteristics of the maximal segment associated with the inflexion zone, the thick red segment is between the straight lines $ax - by = \mu + \frac{a+b}{2}$ and $ax - by = \mu + \frac{a+b}{2} - 1$. The remainders differing by one, the interior of this band does not contain any integer point. The lemma follows. □

Theorem 15. AMLP(C) *is a simple polygon with boundary in* $\mathrm{L}_2(C) \setminus \mathrm{L}_1(C)^\circ$.

Proof. We know already that AMLP(C) is a closed polygonal line (Lemma 11). Lemmata 12–14 guarantee that the restriction of the edges of AMLP(C) in each type of zone is always a single polygonal line in $\mathrm{L}_2(C) \setminus \mathrm{L}_1(C)^\circ$. Taking any two zones on C, say C_{l_1,l_2} and $C_{l'_1,l'_2}$, these contours may share a point if and only if they are consecutive ($l_2 = l'_1$ or $l_1 = l'_2$). Furthermore, since $C \oplus Sq$ is an annulus with two simple polygonal boundaries, the sets $C_{l_1,l_2} \oplus Sq$ and $C_{l'_1,l'_2} \oplus Sq$ may have a non-empty intersection if and only if $l_2 \leq l'_1 \leq l_2 + 2$ or $l_1 - 2 \leq l'_2 \leq l_1$. We remark that an inflexion zone contains at least two points and, if it contains only two, it is surrounded by concave or convex zones, each of which with at least three points. A convex or concave zone may be reduced to one point but is then surrounded by inflexion zones each of which with at least three points. There is thus at most one intermediate zone between two zones with a non-empty intersection.

If there is none, then one of them is an inflexion zone and the other is a convex or a concave zone. At a convex junction C_k, AMLP$(C) \cap (C_k \oplus Sq)$ is reduced to the point $\mathrm{in}(C_k)$, therefore with no other self-intersections. A symmetric result is obtained at a concave junction.

If there is one, this intermediate zone is composed of one, two or three points (0, 1, 2 linels). In all cases, AMLP(C) in this zone is either reduced to a point or to a straight segment. In the former case, the two parts of AMLP(C) coming from the two surrounding zones touch at this point and there is no other self-intersection. In the latter case, the straight segment in the intermediate zone joins the last point of the previous zone to the first point of the next zone and has therefore an otherwise empty intersection with both of them. □

We mention a property of a $\wedge$-turn (a symmetric one exists for $\vee$-turn). We already know that the slopes of the two maximal segments of a $\wedge$-turn are decreasing but we need to be more precise to determine the slope of the AMLP(C) in an inflexion zone.

Lemma 16. *Let M_i and M_{i+1} be two consecutive maximal segments forming a $\wedge$-turn, and $\overrightarrow{c_i}$ and $\overrightarrow{c_{i+1}}$ their quadrant vectors. Setting $A = \mathrm{L}_f(M_i) + \overrightarrow{c_i}$ and $A' = \mathrm{L}_l(M_{i+1}) + \overrightarrow{c_{i+1}}$. Let θ be the direction of the DSS $M_i \cap M_{i+1}$, α be the direction of the vector $\overrightarrow{A\mathrm{U}_l(M_i)}$, α' be the direction of the vector $\overrightarrow{\mathrm{U}_f(M_{i+1})A'}$. Then*

$$\theta_i > \alpha \geq \theta \geq \alpha' > \theta_{i+1}. \tag{2}$$

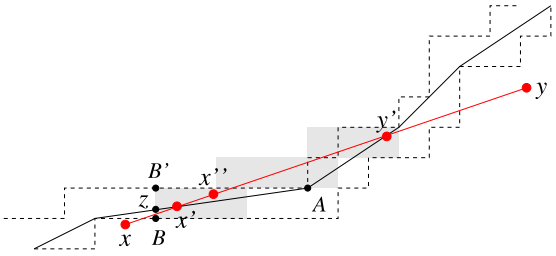


Fig. 6. Illustration of the proof of Lemma 20.

Proof. Let us assume we are in the first quadrant and let us identify the DSS $M_i \cap M_{i+1}$ with its part $C_{k,l}$ on the contour. It is the common part of two consecutive maximal segments, then both C_{k-1} and C_{l+1} are weak lower leaning points. The slope of $C_{k-1,l}$, and thus the slope of M_i, are right descendants of the slope of $C_{k,l}$ in the Stern–Brocot tree of fractions. Since A is a weak upper leaning point of M_i, the slope of the straight line from A to the last upper leaning point of M_i is either a left descendant of the slope of $C_{k-1,l}$ (and is then greater than the slope of its ancestor $C_{k,l}$) or exactly the ancestor of the slope $C_{k-1,l}$ (and is then equal to the slope $C_{k,l}$). We have just proved $\theta_i > \alpha \geq \theta$. The right part is proven similarly. □

Corollary 17. *Let θ be the direction of the edge e of $\partial\mathrm{AMLP}(C)$ in a $(\wedge, \vee)$-inflexion zone. Let θ' and θ'' be the respective directions of the edges of $\partial\mathrm{AMLP}(C)$ just before and just after e. Then $\theta' \geq \theta$ and $\theta \leq \theta''$.*

Proof. Let M_i the maximal segment carrying the $(\wedge, \vee)$-inflexion zone. Applying Lemma 16 to the $\wedge$-turn (M_{i-1}, M_i) gives $\theta' \geq \theta$ (worst case is M_{i-1} is a $(\vee, \wedge)$-zone and we must keep α for M_{i-1} and α' for M_i). Applying the symmetric of Lemma 16 for a $\vee$-turn gives the other part. □

Corollary 18. *Convex vertices of* $\mathrm{AMLP}(C)$ *are inside pixels of C (i.e. $\in \partial\mathrm{L}_1(C)$), concave vertices of* $\mathrm{AMLP}(C)$ *are outside pixels of C (i.e. $\in \partial\mathrm{L}_2(C)$).*

Proof. We already know that in a convex zone of C, vertices of $\mathrm{AMLP}(C)$ are by Definition 10 inside pixels with strictly decreasing edge directions. In a concave zone of C, vertices of $\mathrm{AMLP}(C)$ are outside pixels with strictly increasing edge directions. Around an inflexion zone of direction θ_i, $\theta_{i-1} > \theta_i$ impose a $(\wedge, \vee)$-zone by Corollary 17, which means that the vertex is the inside pixel of an upper leaning point. The reasoning is similar when $\theta_{i-1} < \theta_i$. □

4.4. $\mathrm{AMLP}(C)$ *is the MLP of C*

We can now prove that the polygon $\mathrm{AMLP}(C)$ is the minimum perimeter polygon of $\mathrm{L}_1(C)$, $\mathrm{L}_2(C)$, or the so-called minimum length polygon of C in the terminology of Klette et al. We recalled in Theorem 5 the equivalence of minimum perimeter polygon with relative convex hull. We will therefore prove the following.

Theorem 19. *If C is a simple 4-connected digital contour, then* $\mathrm{AMLP}(C)$ *is the convex hull of* $\mathrm{L}_1(C)$ *relative to* $\mathrm{L}_2(C)$ *or, otherwise said,* $\mathrm{AMLP}(C)$ *is the intersection of every* $\mathrm{L}_2(C)$*-convex set containing* $\mathrm{L}_1(C)$.

Proof. We proceed in four steps (the two first ones are proven afterward):

1. $\mathrm{AMLP}(C)$ is a $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$ (Lemma 20).
2. Every convex and every concave vertex of $\mathrm{AMLP}(C)$ belongs to every $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$ (Lemma 21).
3. Every edge of $\mathrm{AMLP}(C)$ belongs to every $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$. Indeed, let U be such a $\mathrm{L}_2(C)$-convex set and let $\overline{PQ}$ be some edge of $\mathrm{AMLP}(C)$. Now, P and Q belongs to U (from step (2)). As $\overline{PQ} \subset \mathrm{L}_2(C)$, by definition of relative convexity, $\overline{PQ} \subset U$, which concludes.
4. Points (2) and (3) implies $\partial\mathrm{AMLP}(C)$ is included in every $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$, which proves that $\partial\mathrm{AMLP}(C)$ is included in the intersection of every $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$. Let x be some point in $\mathrm{AMLP}(C)^\circ$. Taking any straight line containing x, it intersects $\partial\mathrm{AMLP}(C)$ at least two points by Jordan's theorem. Picking the points closest to x on each side, say P and Q, the same reasoning as in step (3) concludes that the whole segment $\overline{PQ}$ and hence x belong to every $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$. We have proven that $\mathrm{AMLP}(C)$ is included in the intersection of every $\mathrm{L}_2(C)$-convex set containing $\mathrm{L}_1(C)$. Being itself such a relative convex set (point (1)), it is necessarily the convex hull of $\mathrm{L}_1(C)$ relative to $\mathrm{L}_2(C)$. □

We detail now the properties related to steps (1) and (2).

Lemma 20. $\mathrm{AMLP}(C)$ *is a* $\mathrm{L}_2(C)$*-convex set containing* $\mathrm{L}_1(C)$.

Proof. Fig. 6 illustrates this proof. It is clear from Theorem 15 that $\mathrm{AMLP}(C) \subset \mathrm{L}_2(C)$. Furthermore, $\partial\mathrm{AMLP}(C)$ is a deformation of C in the annulus $C \oplus Sq$, bounded on the inside by $\mathrm{L}_1(C)$, and therefore $\mathrm{AMLP}(C) \supset \mathrm{L}_1(C)$.

In order to show that AMLP(C) is a $L_2(C)$-convex, it remains to show that for every $x, y \in$ AMLP(C) with $\overline{xy} \subseteq L_2(C)$ it holds that $\overline{xy} \subseteq$ AMLP(C). Taking the contrapositive of preceding statement, let $x, y \in$ AMLP(C) such that $\overline{xy} \not\subseteq$ AMLP(C), we shall prove that $\overline{xy} \not\subseteq L_2(C)$. Let x' be the element of $\overline{xy} \cap \partial$AMLP($C$) closest to x (existence guaranteed by Jordan theorem). The open ray $(x'y)$ has a connected part $(x'y')$ outside AMLP(C). The sequence S of unit squares in the half-integer plane intersected by $(x'y')$ is itself a digital straight segment, whose first square is $C_i \oplus Sq$ with C_i some point of C. Since $(x'y')$ is not in AMLP(C), it is even less in $L_1(C)^\circ$. Thus the squares of S are outside $L_1(C)^\circ$.

If the ray $(x'y')$ goes outside $L_2(C)$, then we have found a point of $\overline{xy}$ that is not in $L_2(C)$, which concludes the argument. Otherwise we shall reach a contradiction. Indeed the ray has a cover of consecutive unit squares in $L_2(C) \setminus L_1(C)^\circ$. Otherwise said, there is a subpart $C_{i,j}$ of the digital contour such that $C_{i,j} \oplus Sq$ covers $(x'y')$. Without loss of generality, we may assume $\overrightarrow{xy}$ is in the first quadrant, with direction $0 \leq p < \frac{\pi}{2}$. Then the direction of ∂AMLP(C) at x' is smaller than p. It is also necessary that the direction of ∂AMLP(C) at y' is greater than p. It is thus compulsory to have at least one concave vertex strictly between x' and y' along the boundary of AMLP(C). We denote by A the first concave point encountered when moving from C_i to C_j. By Corollary 18, the point A is the outside pixel of some C_k and belongs to $L_2(C)$. Remark also that the square $C_i \oplus Sq$ has at least one side joining a point B on $L_1(C)$ to a point B' on $L_2(C)$ that does not intersect $(x'y')$. The point z at the intersection of $\overline{BB'}$ and ∂AMLP(C) is not between x' and y'.

The curve starting from z to B' then moving along $\partial L_2(C)$ until point A then going back along ∂AMLP(C) until z is a Jordan curve. Since $(x'y')$ goes outside AMLP(C), there is a point $x'' \in (x'y') \cap L_2(C)$ which is inside this curve. But point y' is outside this curve. The segment $\overline{x''y'}$ thus crosses it at some point. It cannot be between z to B' nor between A and z otherwise y would be in-between. It cannot also be along $\partial L_2(C)$ but not at A otherwise then it goes outside $L_2(C)$ (case treated above). If the ray exits exactly at A then the direction p is smaller than the direction θ of ∂AMLP(C) just before A. But the sequence of directions from x' to A is strictly decreasing (A is the first concave point). And the direction of $(x'y)$ is strictly greater than the direction of ∂AMLP(C) at x'. We have just build some sequence $p > \alpha_1 > \cdots > \theta > p$. This is a contradiction. □

Lemma 21. *Every convex and every concave vertex of* AMLP(C) *belongs to every* $L_2(C)$*-convex set containing* $L_1(C)$.

Proof. Let U be a $L_2(C)$-convex set containing $L_1(C)$. Let A be some convex vertex of AMLP(C), and let us prove it belongs to U. By Corollary 18, the point A is the inside pixel of some C_k and belongs to $\partial L_1(C)$. Thus $A \in L_1(C) \subset U$.

Let A' be some concave vertex of AMLP(C). We shall prove that $A' \in U$. Let B_1 be the first convex vertex of AMLP(C) before A' and B_2 be the first convex vertex of AMLP(C) after A'. Let $\mathcal{C}$ be the curve going from B_1 to B_2 following ∂AMLP(C) (so that $A' \in \mathcal{C}$) and going back to B_1 following $\partial L_1(C)$. Lemma 13 ensures that $\mathcal{C}$ is a Jordan curve since a concave part of AMLP(C) does not intersect $L_1(C)$.

Let D be the angle bisector of the two edges of AMLP(C) joining in A'. Let the line D' be the perpendicular to D that intersects it at A'. Since A' is a concave vertex relatively to AMLP(C), it is also concave relatively to $\mathcal{C}$ and so, by Jordan's Theorem, there exist two closest points x, y distinct from A' on this Jordan curve such that $A' \in \overline{xy}$. These two points x and y must be on $\partial L_1(C)$ because the whole part of $\mathcal{C}$ going from B_1 to B_2 along AMLP(C) is concave.

Thus, belonging to $L_1(C)$, points x and y belongs to U. Since $\overline{xy} \subset L_2(C)$, relative convexity implies $\overline{xy} \subset U$, which in turns implies $A' \in U$. □

4.5. The arithmetic MLP can be computed in linear time

We finish by providing an algorithm to compute AMLP(C).

Algorithm 1: Computation of AMLP(C). On line 1 the tangential cover of C is computed using [21] (see boxes on Figure 3, left). On line 2 the (α, β)-zones are computed following Definition 7 (see colors on Figure 3, center). On line 7 the left or right envelope can be computed with for instance Melkman's algorithm [24] (see Figure 3, right).

Input: C: A digital contour
Output: S: A sequence of points, that is the MLP of C

```
1 Compute (M_i)_{i=0..m-1} the tangential cover of C;
2 Decompose (M_i)_{i=0..m-1} in (α, β)-zones (Z_j)_{j=1..z}; // WHERE α, β ∈ {∧, ∨}
3 S = ();
4 for j = 1 to z do
5     if Z_j is a (∧, ∧) or (∨, ∨)-zone then
6         for each quadrant word Q_i of Z_j do
7             add to the back of S the left (if (∧, ∧)-zone) or right (if (∨, ∨)-zone) envelope of Q_i ;
      else
8         add to the back of S the segment defined in Definition 10.
9 return S
```

Theorem 22. *Algorithm* 1 *computes* AMLP(C) *in time linear with respect to the length of* C.

Proof. Let n be the number of points of C. Computation of the tangential cover on line 1 is performed in linear time according to [12] or [21]. The computation on line 2 is clearly proportional to m which is also $O(n)$. Finally, on line 7, there are two cases to consider. Given a convex or concave zone, the AMLP(C) is some convex hull of a simple polygonal line which is computed in time proportional to the length of the zone using [24] or [4]. Given an inflexion zone, the computation is reduced to a segment, thus a $O(1)$ operation. Total computation time is $O(n)$. □

5. Combinatorial definition of the minimum length polygon

Based on the combinatorial characterization of digital convexity obtained in [4], we propose a new algorithmic definition of the minimum length polygon. We begin this section by recalling some standard definitions and useful properties of word combinatorics.

Given an arbitrary ordered alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ with the order $a_1 < a_2 < \cdots < a_n$, written $\mathcal{A} = \{a_1 < a_2 < \cdots < a_n\}$ for short, we extend this order to words over $\mathcal{A}$ using the lexicographic order. We note $|w|_a$ the number of occurrences of the letter a in w and $|w| = \sum_{a\in\mathcal{A}} |w|_a$ is the length of w. Let $\mathcal{A}^n$ be the set of all words of length n over $\mathcal{A}$, in particular $\mathcal{A}^0 = \{\varepsilon\}$ where ε is called the *empty word*. A word w is *non-empty* if $w \neq \varepsilon$ and we note $\mathcal{A}^* = \cup_{n\geq 0} \mathcal{A}^n$ and $\mathcal{A}^+ = \mathcal{A}^*\backslash\{\varepsilon\}$. The i-th letter of a word w is $w[i]$ and we refer to factors of w like this: $w = w[1 : i-1]w[i : i+j]w[i+j+1 : n]$, where $w[1 : i-1]$ is a prefix of w, $w[i+j+1 : n]$ is a suffix of w and all three are factors of w. We denote by $\mathrm{Pref}(w)$ the set of all prefixes of w and $\mathrm{Pref}_+(w)$ the set of all proper prefixes of w that is the set $\mathrm{Pref}_+(w) = \mathrm{Pref}(w) \setminus \{w\}$. Note for any word w, $\varepsilon \in \mathrm{Pref}(w)$.

By reference to the Freeman coding, given a word $w \in \{0, 1, 2, 3\}^n$ the translation vector associated to w is $\overrightarrow{w} = (|w|_0 - |w|_2, |w|_1 - |w|_3)$.

5.1. Lyndon words

Definition 23. A non-empty word w over the ordered alphabet $\mathcal{A}$ is a *Lyndon word* if $w < v$ for any non-empty suffix v of w. $\mathbf{L}_{\mathcal{A}}$ is the set of all Lyndon words over $\mathcal{A}$.

Note that when the alphabet is unambiguous, we might simply write **L**.

Theorem 24 (*[22] Theorem 5.1.1*)**.** *Any non-empty word w over $\mathcal{A}$ admits a unique factorization (called* Lyndon factorization*) as a sequence of decreasing Lyndon words: $w = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ with $l_1 > l_2 > \cdots > l_k$ where $n_i \geq 1$ and $l_i \in \mathbf{L}_{\mathcal{A}}$ for all $1 \leq i \leq k$.*

We define the function FLF, called *first Lyndon factor*, as $\mathrm{FLF}(w, \mathcal{A}) = (l_1, n_1)$ where $w = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ is its unique Lyndon factorization according to the ordered alphabet $\mathcal{A}$. For practical reasons, given a word $w \in \mathcal{A}^+$ such that $\mathrm{FLF}(w, \mathcal{A}) = (u, k)$, we define the auxiliary functions $\mathrm{FLF}_{one}(w, \mathcal{A}) = u$ and $\mathrm{FLF}_{all}(w, \mathcal{A}) = u^k$. If the alphabet is implicit, we may simply write $\mathrm{FLF}(w)$. Finally, we recall the following basic property of Lyndon words.

Property 25 (*[22], Proposition 5.1.3*)**.** *Given $u, v \in \mathbf{L}_{\mathcal{A}}$, if $u < v$ then $uv \in \mathbf{L}_{\mathcal{A}}$ so the inequality $u < uv < v$ holds.*

5.2. Christoffel words

Introduced by Christoffel in [5], Christoffel words where reinvestigated by Borel and Laubie in [3]. Since then their impressive combinatorial structure has been studied by many, see [2] for a comprehensive self-contained survey. Here is one of the many equivalent definitions of Christoffel words.[1]

Definition 26. A *Christoffel* word on the alphabet $\{a < b\}$ is the Freeman code of the path joining two consecutive upper leaning points of a DSS with positive slope according to the convention that the letter a codes an horizontal step and the letter b codes a vertical one.

A Christoffel word is said to be trivial if it has length 1 and we note $\mathbf{C}_{a<b}$ the set of all Christoffel words over the ordered alphabet $\{a < b\}$, while the set of non-trivial Christoffel words is denoted $\mathbf{C}'_{a<b}$. Again, when the alphabet is unambiguous we simply write **C**.

Referring to the Freeman code, the *slope* of w is defined as $\rho(w) = |w|_b/|w|_a$ with the convention that $1/0 = \infty$. In the case of Christoffel words, unlike the general case, the lexicographic order matches the natural order on the slopes: $u, v \in \mathbf{C}_{a<b}$ implies $(u < v \iff \rho(u) < \rho(v))$.

A convex polyomino being composed of only one convex zone, Lemma 8 provides a natural decomposition of its boundary in four quadrant words. Our combinatorial view of convexity is based on the following result which characterizes convex quadrant words.

[1] These words are sometimes referred as *primitive lower Christoffel* words.

Theorem 27 ([4]). *A hv-convex polyomino P is convex if and only if the factorization as decreasing Lyndon words of each quadrant words $Q_{a<b} = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ is such that $l_i \in \mathbf{C}_{a<b}$ for all $1 \leq i \leq n_k$. Moreover, in the case where P is convex, for each quadrant, the edges of its convex hull coincide with the vectors $n_i \overrightarrow{l_i}$.*

In other terms, we have convexity when Lyndon factors are Christoffel words. This leads us to define the variant *First Lyndon Christoffel Word* FLCF of the FLF function as follows. Let w be a word over $\mathcal{A} = \{a_1 < a_2 < a_3 < a_4\}$, such that $w[1] = a_2$ and $\mathrm{FLF}(w, \mathcal{A}) = (u, k)$, then

$$\mathrm{FLCF}(w, \mathcal{A}) = \begin{cases} (u, k, \texttt{true}) & \text{if } u \text{ is in } \mathbf{C}_{a_2<a_3}, \\ (\varepsilon, 0, \texttt{false}) & \text{otherwise.} \end{cases}$$

5.3. Definition of the CMLP

We define the combinatorial minimum length polygon algorithmically using Algorithm 3 which simply computes vertices given by a list of edges given by Algorithm 2 which is base on the function FLCF. We suppose that the word w codes the boundary of a polyomino P starting from the point (x_0, y_0) which is the lowest point among the leftmost points of P (i.e. $x_0 = \min\{x | (x, y) \in P\}$ and $y_0 = \min\{y | (x_0, y) \in P\}$).

Algorithm 2: `nextEdge`

Input: (u, A) such that $u \in A^+$ and $\mathcal{A} = \{a_1 < a_2 < a_3 < a_4\}$.
Output: (x, l, A) with $x \in \mathbb{Z}^2$, $l \in \mathbb{N}$.

```
(v, k, inC) ← FLCF(u, A);   // Try to extract next Christoffel edge.
x = kv⃗;  l = k|v|;
if v = a_2 then
    // Quadrant change.
    A ← {a_4 < a_1 < a_2 < a_3};
    x ← x − v⃗;
else if not inC then
    // Inflexion detected.
    t ← u[0];
    u[0] ← a_3;
    A ← {a_4 < a_3 < a_2 < a_1};
    (x, l, A) ← nextEdge(u, A);
    u[0] ← t;
return (x, l, A)
```

Algorithm 3: Computation of vertices from Algorithm 2.

Input: $w \in \{0, 1, 2, 3\}^N$ the boundary word of P.
Output: $(x_0, x_1, x_2, \ldots)$ a list of vertices that form the CMLP of P.

```
x ← (0, 0);  i ← 0;  A ← {0 < 1 < 2 < 3};
w ← w · 10;  N ← N + 2 ;
while w ≠ ε do
    (v, l, A) = nextEdge(w, A);
    x_{i+1} ← x_i + v ;
    i ← i + 1 ;
    w ← w[l + 1 : N] ;
    N ← N − l ;
return (x_0, x_1, x_2, ..., x_i)
```

In order to illustrate how Algorithm 2 works, we discuss the geometrical interpretation of the modifications performed to the alphabet $\mathcal{A}$ in Algorithm 2. First, notice that the alphabet is initialized by $\mathcal{A} = \{0 < 1 < 2 < 3\}$ as we know that the leftmost part of the shape is convex (of the form $21 \cdots 10$), the contour going clockwise around the shape. All through the algorithm, it shall always be the case that when analyzing a convex quadrant word $Q_{a_2<a_3}$ the alphabet $\mathcal{A}$ is set to $\{a_1 < a_2 < a_3 < a_4\}$ so that the word a_2a_1 codes a quadrant change while a_3a_4 codes a change of convexity type.

A bijective map $\mu : \mathcal{A} \to \mathcal{A}$ over the letters $\mathcal{A}$ extends naturally to any word $w \in \mathcal{A}^n$ as $\mu(w) = \mu(w[1])\,\mu(w[2]) \cdots \mu(w[n])$. Using the notation $\mu(\mathcal{A}) = \{\mu(a_1) < \mu(a_2) < \cdots\}$, clearly $\mu(w) \in \mathbf{L}_{\mathcal{A}} \iff w \in \mathbf{L}_{\mu^{-1}(\mathcal{A})}$

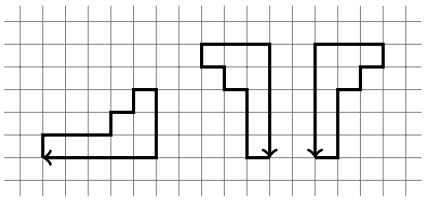


Fig. 7. *Left*. A path coded by $w = 1000101033322222$. *Center*. $r^{-1}(w) = 2111212100033333$ and *Right*. $\overline{w} = 0111010122233333$.

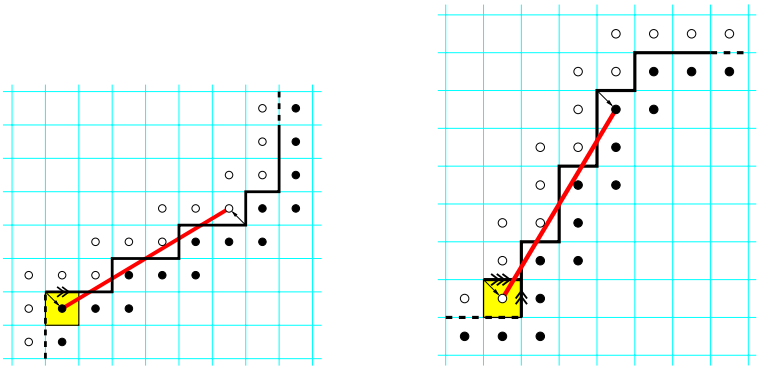


Fig. 8. Illustration of Algorithm 2. *Left*. Starting from the step $\gg$, the path coded by $w = 001001001011\cdots$ is such that FLCF$(w, \{3 < 0 < 1 < 2\}) = (\varepsilon, 0, \texttt{false})$. *Right*. Starting from the step $\gg$, the path coded by $\overline{w} = 110110110100\cdots$, and starting from the step $\ggg$, the path coded by $f(w) = 010110110100\cdots$ is such that FLCF$(f(w), \{3 < 0 < 1 < 2\}) = (01011011, 1, \texttt{true})$. The thick red line shows the MLP's edge obtained from 01011011. This edge defines an inflexion zone, more precisely a $(\wedge, \vee)$-zone. Working with $f(w)$ allows to consider both extremities of the inflexion zone as inside pixels. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

and $\mu(w) \in \mathbf{C}_{a<b} \iff w \in \mathbf{C}_{\mu^{-1}(a)<\mu^{-1}(b)}$. Algorithm 2 uses this fact so that instead of applying transformations to the whole word w, only the order relation over the four letter alphabet is changed. Let w be the contour word of C over $\mathcal{A} = \{3 < 0 < 1 < 2\}$ and define $r : \mathcal{A} \to \mathcal{A}$ as $r(3) = 2, r(0) = 3, r(1) = 0$ and $r(2) = 1$. One verifies that the contour coded by $r^{-1}(w)$ corresponds to a rotation by $\pi/2$ (see Fig. 7). This explains line 4 of Algorithm 2 which is called when a quadrant change occurs.

Similarly, let $\mathcal{A} = \{a_1 < a_2 < a_3 < a_4\}$ and define the *bar* operator $\overline{(\,)}$ as $\overline{a_1} = a_4, \overline{a_2} = a_3, \overline{a_3} = a_2$ and $\overline{a_4} = a_1$. Consider a quadrant word $Q_{a<b}$, one verifies that $\overline{Q_{a<b}}$ corresponds to a reflexion by the line $y = x$ if $\{a, b\} \in \{\{0, 3\}, \{2, 1\}\}$ or by the line $y = -x$ if $\{a, b\} \in \{\{1, 0\}, \{3, 2\}\}$. Once again, see Fig. 7 for an example. Roughly speaking, this transformation turns this part of the contour *inside out*, so that computing the left envelope of $Q_{a<b}$ is equivalent to the computation of the right envelope of $\overline{Q_{a<b}}$. This explains line 9 of Algorithm 2.

The modification of the first letter of w at line 8 is due to the fact that w code the inter-pixel path. Since the condition at line 6 detects a change in the convexity type, the inner pixel adjacent to the step coded be $w[1]$ must now be consider as an outside pixel (see Fig. 8). This is done by switching the value of the first letter of w from a_2 to a_3. An equivalent point of view is given by the following involution on words. Let $w = au$ be a non-empty word with $a \in \mathcal{A}$ and $u \in \mathcal{A}^*$, we define the function f by

$$f(au) = a\overline{u}. \tag{3}$$

Definition 28. The *combinatorial MLP* of a digital contour C, noted CMLP(C), is obtained by joining consecutive vertices given as output of Algorithm 3.

We suppose that the lowest pixel among the leftmost pixels of the polyomino P is centered at $(0, 0)$. This ensures that point $v_0 = (0, 0)$ is a vertex of $CMLP(C)$. Moreover, in order to close the polygonal path computed, the word 10 is added at the end of w so that an extra vertex located at $(0, 0)$ is added at the end of the list closing the polygonal line (line 2 in Algorithm 3).

5.4. The CMLP is the same as the AMLP

The main result in order to show that CMLP and AMLP are both the same polygon is given by Theorem 27. Indeed, it implies that for a given convex quadrant word, Algorithm 3 computes its convex hull. Now we need to show that these parts are all connected with each other correctly.

Lemma 29. *Let $M = C_{i,j}$ be a $(\wedge, \vee)$-MDSS corresponding to an inflexion zone of C, and let the word $w \in \mathcal{A}^*$ be the Freeman code of C starting from $\mathrm{U}_f(M)$. The couple $(c, k) = \mathrm{FLF}(f(w), \mathcal{A})$ is such that $\overrightarrow{c} = \mathrm{out}(\mathrm{L}_l(M)) - \mathrm{in}(\mathrm{U}_f(M))$.*

Proof. Let M be a $(\wedge, \vee)$-MDSS. Moreover, without lost of generality, consider the case where the quadrant vector is $(-1/2, 1/2)$ and $\mathcal{A} = \{3 < 0 < 1 < 2\}$.

Starting from the DSS M, we use the algorithmic techniques from [21] in order to remove points at the beginning of the DSS one by one until $\mathrm{U}_f(M)$ is removed. After that, we use technique from [7] in order to add the point $A = \mathrm{U}_f(M) + (1, -1)$. Let k be such that $C_k = \mathrm{U}_f(M)$ consider the DSS $M' = \{A\} \cup C_{k+1,j}$.

The proof is made in two parts. First we show that $A = \mathrm{L}_f(M')$ and that $\mathrm{L}_l(M)$ is the second lower leaning point of M'. Then we show that the word $c = \mathrm{FLF}_{one}(f(w))$ is defined by those two leaning points and satisfies $\overrightarrow{c} = \mathrm{out}(\mathrm{L}_l(M)) - \mathrm{in}(\mathrm{U}_f(M))$.

For the first part, there are two cases to consider, depending if $C_{k+1,j}$ has same slope as M or not.

(a) If $C_{k+1,j}$ has same slope as M, then $\mathrm{U}_f(M)$ does not change the slope of $C_{k+1,j}$ when added, and it is an upper leaning point. Then point A, being its translation by $(1, -1)$ is necessarily a lower weakly exterior point to the digital straight line supporting $C_{k+1,j}$. Therefore $M' = \{A\} \cup C_{k+1,j}$ has a greater slope than $C_{k+1,j}$. Algorithm [7] tells that A is its first lower leaning point ($A = \mathrm{L}_f(M')$) while the last lower leaning point remains unchanged ($\mathrm{L}_l(M) = \mathrm{L}_l(C_{k,j}) = \mathrm{L}_l(M')$). We can conclude since there are only two leaning points.

(b) If the slopes of $C_{k+1,j}$ and M are different, then $\mathrm{U}_f(M)$ is an upper weakly exterior point to $C_{k+1,j}$. Then point A, being its translation by $(1, -1)$ is necessarily a lower leaning point to the digital straight line supporting $C_{k+1,j}$, and we have $A = \mathrm{L}_f(M')$. Let B be the second lower leaning point of M', then B is the first lower leaning point of $C_{k+1,j}$. Algorithm [7] tells that $C_{k,j}$ has B as its last lower leaning point. Since M and $C_{k,j}$ have the same last lower leaning points, we conclude $\mathrm{L}_l(M) = B$.

Let R be the reflexion of M' by the line $y = x$. Of course, R is a DSS whose upper leaning points are the lower leaning points of M'. Call U the image of A in R and U' the image of $\mathrm{L}_l(M)$. From what precedes, it is clear that UU' is a Christoffel word, that we denote by c. By construction, we have

$$\overrightarrow{c} = \mathrm{L}_l(M) - (\mathrm{U}_f(M) + (1, -1)) = \mathrm{out}(\mathrm{L}_l(M)) - \mathrm{in}(\mathrm{U}_f(M)).$$

Let v be the Freeman code of the DSS R. As R starts on an upper leaning point, there exists some positive integer k and word p proper prefix of c such that $v = c^k p$. Since the DSS M is not extensible to the right, the following letter in w is either a_3 or a_4. In the reflexion $f(w)$, this letter, say a, is either a_2 or a_1. Using Lemma 39 (see the Appendix), with $u = c$, we conclude that $\mathrm{FLF}_{one}(f(w)) = c$. □

The next lemma has a symmetric proof.

Lemma 30. *Let $M = C_{i,j}$ be a $(\vee, \wedge)$-MDSS (resp. $(\vee, \wedge)$-MDSS) corresponding to an inflexion zone of C, and let the word $w \in \mathcal{A}^*$ be the Freeman code of C starting from $\mathrm{L}_f(M)$. The couple $(c, k) = \mathrm{FLF}(f(w), \mathcal{A})$ is such that $\overrightarrow{c} = \mathrm{in}(\mathrm{U}_l(M)) - \mathrm{out}(\mathrm{L}_f(M))$.*

Theorem 31. *AMLP(C) and CMLP(C) are the same polygon.*

Proof. First of all, Algorithm 3 starts at the lowest point among the leftmost points of the shape. It starts thus exactly at the end of some word in $Q_{1<2}$. The first letter is clearly 1 so the starting alphabet is consistent for FLCF. The vertical leftmost edge is thus extracted and is followed by a quadrant change. After that, we are now in some quadrant word $Q_{0<1}$ with the alphabet $\{3 < 0 < 1 < 2\}$ in a convex zone. Again, the first letter is 0 so the alphabet is consistent for FLCF.

By Theorem 27, on any convex part that stays in the same quadrant, the edges computed by Algorithm 2 are the same as those of the convex hull, which is the left envelope of this part. Quadrant changes are managed by the condition at line 3 of Algorithm 2. In the case of a quadrant change, the alphabet is modified in order to correspond to the new quadrant, and thus the following letter will be consistent for the next FLCF. Note that the extracted vector is shortened by 1 (line 5) since the interpixel path is always one step longer than the boundary of $\mathrm{L}_1(C)$ in this case.

On a concave part, the reversal of the alphabet at line 9 of Algorithm 2 reverses the perspective and allows to use exactly the same algorithm to compute the convex hull of the outside pixels of C, which is the right envelope of this part. In such case, the first letter is modified (line 8) so as to be consistent with the next FLCF.

Finally, Lemmas 29 and 30 ensure that the edges computed over the inflexion zones between convex and concave zones, which are detected at line 6 of Algorithm 2, are the same as those of the AMLP(C). □

6. Implementation of FLCF and linear computational complexity of *CMLP*

In order to compute the function FLCF one may use Duval's algorithm [11] which computes the pair $(u, k) = \mathrm{FLF}(w, \mathcal{A})$ for any word w with a time complexity of $O(k|u|)$. This optimal algorithm is attributed to [13] in [23]. In Algorithm 2 we compute $(u, k) = \mathrm{FLF}(w, \mathcal{A})$ but, in the case where $u \notin \mathbf{C}$ the specific output (u, k) does not matter. Based on this idea, we propose a modified version of Duval's algorithm. Algorithm 4 has the ability to determine dynamically if the word read might lead to a Christoffel word and the computation immediately stops if not.

We first present the algorithm computing FLCF. Then, after recalling some helpful combinatorial properties of Christoffel words, we prove the validity of this algorithm and we determine the computational complexity of the whole *CMLP* algorithm.

6.1. The modified Duval's algorithm

Starting from Duval's algorithm, we add a few lines in order to stop the computation on the case where the word read is not a Christoffel word. In Algorithm 4, by considering only the lines tagged by (D), one gets the well known Duval algorithm which computes the FLF function in a time linear in the length of its output (see [23], algorithm `LyndonFactorization`). The extra variables p and q are related to properties of Christoffel words and will be further explained in the next subsections.

By updating these two variables p and q when needed, this new algorithm computes the function FLF exactly as Duval's algorithm if the output is a Christoffel word. Otherwise, it stops at line $(*)$. The test performed to reach line $(*)$ is directly based on the upcoming Corollary 34. Fig. 9 illustrates this algorithm on a simple example.

In addition to this, we also added some optional lines tagged by (CF). In the case where $\mathrm{FLF}_{\mathrm{one}}(w, \mathcal{A}) = c \in \mathbf{C}_{\mathcal{A}}$, the list $[u_0; u_1, \ldots, u_d]$ is the continued fraction of the slope of c. Note that this extra information is not required in Algorithm 2: this is why we do not output it in the context of CMLP computation. One may easily check that this does not affect the overall time complexity of the algorithm.

Algorithm 4: Duval++

Input: $(w, \{a_1 < a_2 < a_3 < a_4\})$ where $w \in \{a_2\} \cdot \{a_1, a_2, a_3, a_4\}^{N-1}$.
Output: (u, k, b) where b is a boolean such that $b \iff (u, k) = \mathrm{FLF}(w, \mathcal{A})$.

```
(D)  i ← 1; j ← 2;
     p ← 1; q ← 2;
(CF) m ← 0; d ← 0; u_d ← 0;
 (D) while j ≤ N and w[i] ≤ w[j] do
 (D)     if w[i] = w[j] then
             if j = q then
                 q ← q + p;
(CF)             m ← m + 1;
 (D)         i ← i + 1;
 (D)     else
             if j ≠ q or w[j] ≠ a_3 then
 (*)             return (ε, 0, false);
 (D)         else
 (D)             i ← 1;
                 q ← 2q − p ;
                 p ← j;
(CF)             if m = 0 then
(CF)                 u_d ← u_d + 1;
(CF)             else
(CF)                 d ← d + 1; u_d ← m;
(CF)                 d ← d + 1; u_d ← 1;
(CF)                 m ← 0
 (D)     j ← j + 1;
 (D) return (w[1 : j − i], ⌊(j−1)/(j−i)⌋, true);
```

6.2. More about Christoffel words

Introduced by Borel and Laubie [3], the *standard factorization* of Christoffel words is the word combinatorics equivalent of the well known *splitting formula* of DSS (see [35]).

Property 32 ([3]). *For all $w \in \mathbf{C}'_{a<b}$, there exist a unique pair $x, y \in \mathbf{C}_{a<b}$ such that $w = xy$. This is called the* standard factorization *w, it is denoted by $w = (x, y)$.*

Note that in such case, the inequality $x < w < y$ holds. As mentioned previously, Christoffel words may be defined in many ways. Here is another characterization describing their internal structure. First, let us define the functions G and D as follows:

$$G, D : \mathbf{C}' \longrightarrow \mathbf{C}'$$
$$G(u, v) = (u, uv),$$
$$D(u, v) = (uv, v).$$

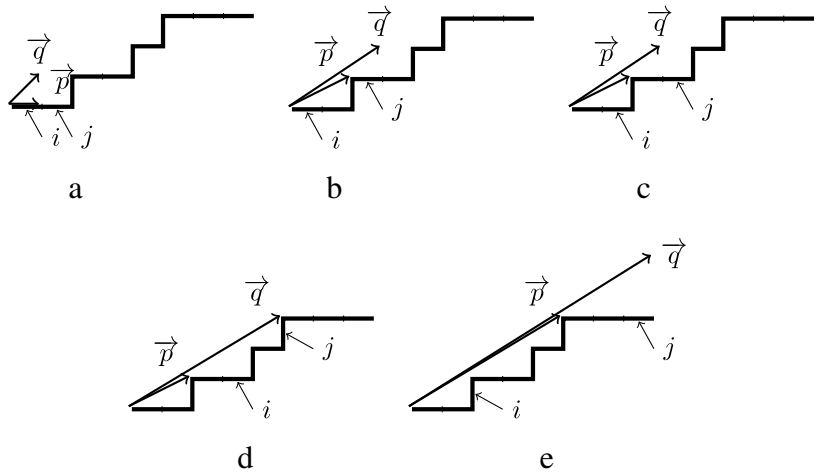


Fig. 9. Illustration of Algorithm 4 on input word 00100101000. At initialization (a): $i = 1, j = 2$, vector $\overrightarrow{p}$ illustrates the Christoffel word $w[1:p]$ and vector $\overrightarrow{q}$ illustrates the next Christoffel word to be obtained if the slope increases. At the beginning of the third iteration (b): $i = 1, j = 4, p = 3, q = 5$, $\overrightarrow{p} = (2, 1)$ and $\overrightarrow{q} = (3, 2)$. A new Christoffel word 001 has been detected at the previous iteration and the algorithm will now test if it is repeated. This is performed by comparing the letters of w starting from position j with the letters starting from position $i = 1$. Also, q has been set to 5 because, according to Corollary 34, a Christoffel word that begins with 001 must have a prefix of the form $(001)^n01$ with $n \geq 1$ so that the shortest one is 00101 with length 5. At the fourth iteration, (c): $i = 2, j = 5, p = 3, q = 5$. Since $j = q$ and $w[i] = w[j]$, the word 00101 is not a prefix of w and the only way the algorithm does not stop is if w starts with at least two copies of $w[1:p] = 001$, q is updated consequently to 8 and the next Christoffel word expected is now 00100101. At the seventh iteration (d): $i = 5, j = 8, p = 3$ and $q = 8$. This time, $w[i] \neq w[j]$ and $j = q$ which means that the expected Christoffel word 00100101 has been read, p and q are updated consequently to 8 and 13 respectively. Also, i is reset to 1 since the algorithm will now check if this new Christoffel word is repeated. Finally, at the tenth iteration, (e): $i = 3, j = 11, p = 8$ and $q = 13$. This algorithm stops since $w[i] > w[j]$. The tuple returned is simply the vector $\overrightarrow{p}$ with the number of repetitions, in this case 1, and `true` since this is locally convex.

Theorem 33 ([3,1])**.** *Every non-trivial Christoffel word over the alphabet $\{a < b\}$ admits a unique sequence $H_1, H_2, \ldots, H_k \in \{G, D\}$ such that*

$$w = H_1 \circ H_2 \circ \cdots \circ H_k(a, b).$$

Reciprocally, for every sequence $H_1, H_2, \ldots, H_k \in \{G, D\}$, the word $w = H_1 \circ H_2 \circ \cdots \circ H_k(a, b)$ is a Christoffel word.

These functions (G for left and D for right) build a tree that is equivalent to the Stern–Brocot tree of irreducible fractions, each fraction being the slope of the corresponding Christoffel word.

This theorem constrains the structure of the prefixes of a Christoffel word as follows.

Corollary 34. *For any $w = (u, v) \in \mathbf{C}'_{a<b}$, let $\chi_w = \{wz \in \mathbf{C}'_{a<b} \mid z \in \{a, b\}^+\}$. One has that*

$$X \in \chi_w \Rightarrow \exists n \geq 1, \qquad w^n v \in \mathrm{Pref}(X). \tag{4}$$

Proof. Any Christoffel word having w as a prefix is in the right subtree of w, i.e. the right subtree of (u, v) (in the tree induced by G and D). Any node of this subtree has a prefix of the form $w^n v$, which concludes. □

Note that $\chi_a = \mathbf{C}'_{a<b}$ and $\chi_b = \emptyset$.

We can now establish some combinatorial properties of Christoffel words which will be used later on to show the correctness of Algorithm 4. Recall that a *period* p of a word w is a positive integer such that $w[i] = w[i + p]$ for all $1 \leq i \leq |w| - p$.

Property 35. *Let $w \in \mathbf{C}'_{a<b}$, then there exist u, x, y such that $w = aub = (x, y)$ with the following properties:*

(a) $a\overline{u}b \in \mathbf{C}_{a<b}$.
(b) *u is a palindrome.*
(c) *$y = y'b$ implies that $y'a \in \mathrm{Pref}(w)$.*

Proof. (a) It is a direct consequence of [2, Theorem 6.3], where the result is attributed to [8].
(b) See for instance [2, Proposition 4.2].
(c) The case $y' = \varepsilon$ is trivial while otherwise, we have $y = y'b = au'b$ and by (b) both words u and u' are palindromes. This means that u' is prefix and suffix of u and so $y' \in \mathrm{Pref}(au) \subset \mathrm{Pref}(w)$. Finally, if $y'a \notin \mathrm{Pref}(w)$ it means that $y'b = y \in \mathrm{Pref}(w)$ and so $y < w$ which contradicts Property 32. □

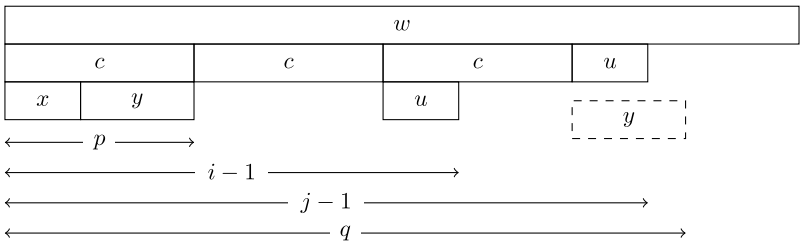


Fig. 10. Illustration of the four conditions of Lemma 36 with $m = 2$ and $\alpha = 2$.

6.3. Validity and computational complexity of algorithm Duval++

In this subsection, we show that algorithm Duval++ (Algorithm 4) computes FLCF and that its time complexity is linear with the size of its output (Proposition 37). Before proving it, we list the formal interpretations of each variable involved in this algorithm (Fig. 10):

j: the index of the new letter in the input word, which will be read during this iteration;
i: the index of the letter that is one period behind the letter at index j;
p: the period of $w[1 : j-1]$, it is also the greatest index smaller then j such that $w[1 : p]$ is a Christoffel word;
q: the expected position of the next upper leaning point, it is also the smallest index greater or equal to j such that $w[1 : q]$ may be a Christoffel word.

The following lemma formalizes these ideas.

Lemma 36. *In Algorithm* 4, *each time the* while *condition is tested, either the word $w[1 : j-1]$ is written only on one letter (a_2 or a_3), or the four following conditions hold:*

(1) $w[1 : p] = c \in \mathbf{C}'_{a_2<a_3}$,
(2) $w[1 : i-1] = c^\alpha \cdot u$ *where* $\alpha \geq 0$, $u \in \mathrm{Pref}(c)$ *and* $u \neq c$,
(3) $w[1 : j-1] = c^{\alpha+1} \cdot u$,
(4) $q = (m+1)|c| + |y|$ *where* $c = (x, y)$, $m \geq 0$ *and* $m|c| + |y| \leq j - 1 < (m+1)|c| + |y|$.

Proof. First of all, one may easily check that starting from the value 2, the variable j is incremented exactly once at every turn of the loop. Also, one verifies that if the letter $w[j]$ is a_1 or a_4 then the algorithm immediately stops. We denote by n the current index of the iteration, starting from 1. We use the convention that $\mathbf{x}_n$ refers to the value of some variable $\mathbf{x}$ at the beginning of iteration n. It is clear that $j_n = n + 1$.

We begin by treating the case where $w[1 : n]$ is written only on one of the letters a_2 or a_3. The only non-trivial case is $w[1 : n] = a_2^n$ and $w[n+1] = a_3$. In such case, after one iteration, we have the following situation:

$$j_{n+1} = n + 2, \qquad w[1 : n+1] = a_2^n a_3, \qquad i_{n+1} = 1, \qquad p_{n+1} = n + 1, \qquad q_{n+1} = 2n - 1,$$

which satisfies the four conditions, with $c_{n+1} = a_2^n a_3 = (a_2, a_2^{n-1} a_3)$, $u = \varepsilon$, $\alpha = 0$, $m = 0$.

We now proceed by induction on n. The case $n = 1$ induces a one-letter word $w[1 : 1]$ and is proved by the preceding argument. Assuming the preceding properties are satisfied for an arbitrary $n > 1$, we prove they hold also for $n + 1$.

There are only three different branches in this loop. To prove the induction, we analyze the three scenarios that might occur at the n-th iteration.

1. $w[i_n] = w[j_n]$ and $j_n < q_n$ (within a periodic pattern, slope is unchanged).
   In this case we have

   $$i_{n+1} = i_n + 1, \qquad j_{n+1} = j_n + 1, \qquad p_{n+1} = p_n, \qquad q_{n+1} = q_n.$$

   It is trivial that condition (1) still holds. On the other hand, letting $a = w[j_n]$, we have

   $$w[1 : i_{n+1} - 1] = c_n^{\alpha_n} u_n a,$$
   $$w[1 : j_{n+1} - 1] = c_n^{\alpha_n + 1} u_n a.$$

   Since $w[j_n]$ is also $w[i_n]$, we have $u_n a \in \mathrm{Pref}(c_n)$. This means that either $\alpha_{n+1} = \alpha_n$ and $u_{n+1} = u_n a \in \mathrm{Pref}(c_{n+1})$ satisfying $u_{n+1} \neq c_{n+1}$, or $\alpha_{n+1} = \alpha_n + 1$, and $u_{n+1} = \varepsilon$. This means that (2) and (3) hold. For condition, (4), j has not increased so much as to modify the value of m since $j_n < q_n$, thus $m_{n+1} = m_n$.
2. $w[i_n] = w[j_n]$ and $j_n = q_n$ (one more reversed pattern, slope is unchanged).

In this case we have

$$i_{n+1} = i_n + 1, \qquad j_{n+1} = j_n + 1, \qquad p_{n+1} = p_n, \qquad q_{n+1} = q_n + p_n.$$

Conditions (1)–(3) are shown exactly the same way as in the previous case. One easily checks that condition (4) still hold with $m_{n+1} = m_n + 1$.

3. $w[i_n] = a_2$, $w[j_n] = a_3$ and $j_n = q_n$ (slope is increased).
In this case, we have

$$i_{n+1} = 1, \qquad j_{n+1} = j_n + 1, \qquad p_{n+1} = j_n, \qquad q_{n+1} = 2q_n - p_n.$$

Since $w[1 : n+1] = w[1 : p_{n+1}] = c_n^{\alpha_n+1} u_n a_3$ with $y_n = u_n a_3$ and $u_n a_2 \in \mathrm{Pref}(c_n)$, Lemma 40(1) (see the Appendix) applies. Thus we have that

$$c_{n+1} = c_n^{\alpha_n+1} u_n a_3 = (c_n, c_n^{\alpha} u_n a_3) = (x_{n+1}, y_{n+1}) \in \mathbf{C}_{a_2<a_3},$$

so condition (1) is satisfied.
Conditions (2) and (3) trivially hold with $\alpha_{n+1} = 0$ and $u_{n+1} = \varepsilon$.
Finally, condition (4) holds with $m_{n+1} = 0$ since

$$q_{n+1} = 2q_n - p_n = 2|c_{n+1}| - |c_n| = |c_{n+1}| + |c_n^{\alpha_n+1} y_n| - |c_n| = |c_{n+1}| + |y_{n+1}|,$$

and $|y_{n+1}| \le j_{n+1} - 1 < |c_{n+1}| + |y_{n+1}|$. □

Using these invariants, we may now prove the validity and linearity of Algorithm 4.

Proposition 37. *Let $w \in \mathcal{A}^n$ where $\mathcal{A} = \{a_1 < a_2 < a_3 < a_4\}$, w is a contour word with first letter a_2, and let $(u, k) = \mathrm{FLF}(w, \mathcal{A})$. We have exactly the two following cases:*

(i) *If $u \in \mathbf{C}_{a_2<a_3}$ then a call to Algorithm 4 with $(w, \mathcal{A})$ as input will return $(u, k, \mathtt{true})$ with time complexity $O(k|u|)$.*
(ii) *If $u \notin \mathbf{C}_{a_2<a_3}$ then a call to Algorithm 4 with input $(w, \mathcal{A})$ will return $(\varepsilon, 0, \mathtt{false})$ while a call to Algorithm 4 with input $(f(w), \mathcal{A})$ will return $(u', k', \mathtt{true})$ where $(u', k') = \mathrm{FLF}(f(w), \mathcal{A})$ and $u' \in \mathbf{C}_{a_2<a_3}$. Moreover, both calls have time complexity $O(k'|u'|)$.*

Proof. First, suppose that Algorithm 4 returns a triplet of the form $(u, k, \mathtt{true})$. By the validity of the original Duval's algorithm, it must be that $\mathrm{FLF}(w, \mathcal{A}) = (u, k)$ so all that remains to see is that $u \in \mathbf{C}_{a_2<a_3}$.

In Lemma 36, conditions (2) and (3) implies that $j - i = p$ at the end of each iteration and so, by condition (1) of the same lemma, $u = w[1 : p] \in \mathbf{C}_{a_2<a_3}$.

In such case, one checks that the time complexity of the algorithm is linear with the length of its output u^k.

On the other hand, suppose that Algorithm 4 returns the output $(\varepsilon, 0, \mathtt{false})$. We show that in such case, $\mathrm{FLF}_{one}(w, \mathcal{A}) \notin \mathbf{C}_{a_2<a_3}$.

Let $\mathbf{x}_f$ be the value of the variable $\mathbf{x}$ when the algorithm reached the *return* on line $(*)$, also let $l = \mathrm{FLF}_{one}(w, \mathcal{A})$ and $l' = \mathrm{FLF}_{one}(f(w), \mathcal{A})$.

Let us first remark that $w[j_f] \neq a_1$, since in this case the algorithm naturally stops due to the lexical ordering (case above). Clearly, if $w[j_f] = a_4$ then l contains the letter a_4 and $l \notin \mathbf{C}_{a_2<a_3}$. The only other possibility is

$$l[i_f] = a_2, \qquad l[j_f] = a_3 \quad \text{and} \quad j_f \neq q_f.$$

In such case, by Lemma 36, the word $w[1 : j_f]$ has the following form

$$w[1 : j_f] = c^{\alpha+1} u a_3, \quad \text{where } c \in \mathbf{C}_{a_2<a_3},\ \alpha \ge 0,\ ua_2 \in \mathrm{Pref}(c).$$

By Lemma 40(3), we have that $l \notin \mathbf{C}_{a_2<a_3}$. Moreover, let $(u', k') = \mathrm{FLF}(f(w), \mathcal{A})$, Lemma 40(3) also ensures that $u' \in \mathbf{C}_{a_2<a_3}$ and $k'|u'| \ge |w[1 : j_f]|/2 = j_f/2$.

Finally, the time complexity of the call to Algorithm 4 with input $(w, \mathcal{A})$ is $O(j_f)$ which is also a $O(k'|u'|)$, while the time complexity of the call to Algorithm 4 with input $(f(w), \mathcal{A})$ is $O(k'|u'|)$ according to (i). □

6.4. Computational complexity of CMLP algorithm

We can now complete our analysis of the time complexity of the algorithm computing the CMLP.

Theorem 38. *Algorithm 3 computes the CMLP of a polyomino in linear time with respect to the size of its contour word.*

Proof. Let w be the contour word of the polyomino with $|w| = N$. We assume first that a call to `nextEdge` has a complexity in $O(l)$ (when its output is $(x, l, \mathcal{A})$). In this case, the overall complexity of Algorithm 3 is $O(N)$. Indeed, all other operations within the loop at line 3 are in constant time (the copy of w is only a renumbering). Then w is shortened by l (lines 7 and 8) at each iteration until it is reduced to the empty word. It is thus obvious that these complexities sum to the length of the input word, which is N.

We prove now that `nextEdge` has a complexity in $O(l)$. We already showed in the proof of Theorem 31 that the first letter of w when calling `nextEdge` is a_2. Proposition 37 thus holds. In case (i) of this proposition, $k|u| = l$ and we conclude. In case (ii) of this proposition, the seemingly recursive call to `nextEdge` has only depth 1 since $u' \in \mathbf{C}_{a_2<a_3}$. We have $k'|u'| = l$ and the total complexity is $O(l) + O(l)$, which concludes the argument. □

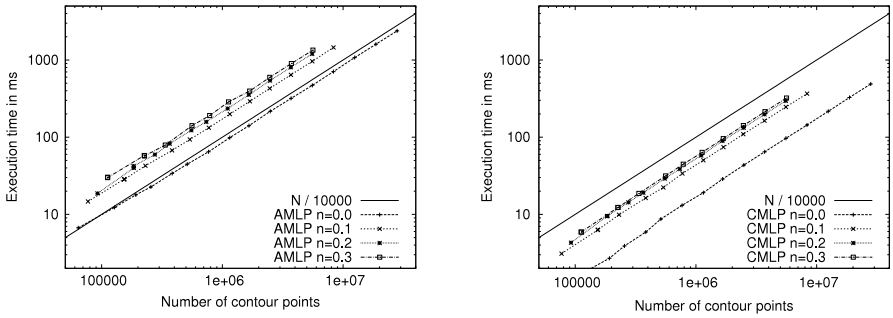


Fig. 11. Computation times of AMLP and CMLP as a function of the size N of the shape contour. For both algorithms, the input shape is an ellipse which may be perturbed by some noise ($n = 0$, no noise, otherwise n denotes the probability to flip a pixel around the initial contour). Both algorithms are clearly linear. The CMLP is between 4 and 5 times faster than the AMLP in all cases. Experiments were performed on an Intel(R) Core(TM)2 Duo 3.33 GHz.

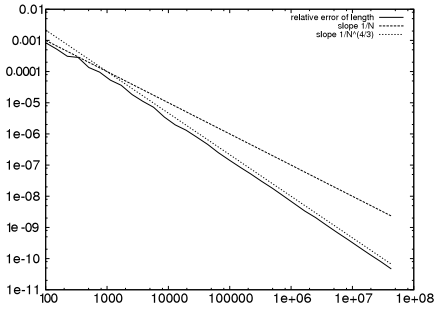


Fig. 12. Perimeter estimation of a digital disk with its MLP. We have plotted the relative error between the length of the MLP and the perimeter of the Euclidean disk. The parameter N is the resolution for sampling the disk.

7. Experimentations and concluding remarks

We have presented two different definitions for the minimum length polygon of a digital contour: one based on an arithmetic approach, the other based on a combinatorial one. Both are shown to be the unique MLP of the contour, and linear time algorithms to compute them are provided. Even though we did not prove it, our notion of MLP has no problem dealing with one pixel wide areas, i.e. holes or bars of one pixel wide, as illustrated in Figs. 13 and 14.

We have implemented both algorithms in C++ and both are available at [16].[2] We have run experiments on various shapes with increasing size and noise level, so as to estimate the respective execution speeds of these algorithms (see Fig. 11). As expected, they are both linear in the size of the contour word. However, the constant is smaller for the CMLP (about 5 times better). As one can see, the CMLP takes about 17.5 ns per contour point on this computer.

As mentioned in the introduction, the MLP of a digital shape is a good perimeter estimator of the underlying Euclidean shape [19,6] and is proved to be multigrid convergent in $O(h)$ for digitization of convex shapes, where $h = 1/N$ is the grid step [18,31,32]. We have checked this property on finer and finer digitizations of a Euclidean disk. Fig. 12 shows that this estimator is indeed convergent but with a speed even faster than the theoretical bound. This discrepancy probably comes from arithmetic properties of smooth shape digitizations [20].

Finally, note that one could modify the given algorithms in order to remove aligned points. On the other hand, one might prefer to keep those redundant points in order to rebuild the original polyomino.

Appendix. Technical lemmata

Lemma 39. *Given a word $w \in \{a_2 < a_3\}^*$ and two letter $a \in \{a_2, a_3\}$ and $b \in \mathcal{A} = \{a_1 < a_2 < a_3 < a_4\}$ such that $w = u^k p$ with $k \geq 1$, $u \in \mathbf{L}_{\mathcal{A}}$, $pa \in \mathrm{Pref}_+(u)$ and $pb \notin \mathrm{Pref}(u)$, then given $z = wbs$ for any suffix $s \in \mathcal{A}^*$:*

1. *if $a > b$, then $\mathrm{FLF}(z, \mathcal{A}) = (u, k)$,*
2. *if $a < b$, then $|\mathrm{FLF}_{one}(z, \mathcal{A})| \geq |u^k pb|$.*

Proof. Using Property 25, one can compute the unique Lyndon factorization of a word z the following way:

– Write z as a list $(l_1, l_2, \ldots, l_n)$ of Lyndon words such that $l_1 l_2 \cdots l_n = w$. This is always possible since all letters are Lyndon words.

[2] Use the executable `freeman2polygon` with option `-dCMLP` for CMLP and `-dCP` for AMLP.

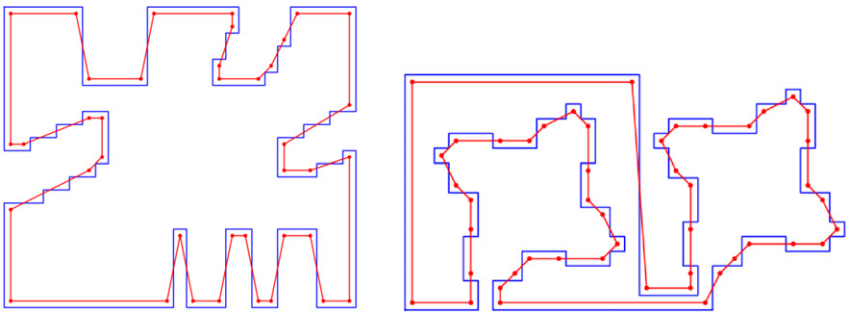


Fig. 13. Examples of MLP.

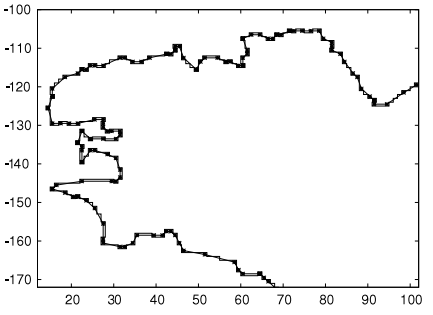


Fig. 14. Extract of the MLP of a digitized France. The shown area is Brittany.

- If $l_i < l_{i+1}$ for some i, then replace the pair (l_i, l_{i+1}) by $(l_i l_{i+1})$ so that the length of the list is reduced by one.
- Repeat the previous step until no such i can be found.

Consider the following factorization of $z = ws$:

$$z = (\underbrace{u, \ldots, u}_{k \text{ times}}, p_1, p_2, \ldots, p_{n_p}, b, s_1, s_2, \ldots, s_{n_s}) \tag{A.1}$$

where $(p_1, p_2, \ldots, p_{n_p})$ is the Lyndon factorization of p and $(s_1, s_2, \ldots, s_{n_s})$ is the Lyndon factorization of s. Since all words in this factorization are Lyndon words, the previous algorithm can be used to compute the Lyndon factorization of w starting from there.

Since $p_1 \in \mathrm{Pref}(p)$ and $p \in \mathrm{Pref}(u)$, the following inequalities hold:

$$u \geq p_1 \geq p_2 \geq \cdots \geq p_{n_p} \quad \text{and} \quad s_1 \geq s_2 \geq \cdots \geq s_{n_s}.$$

On the first iteration of the algorithm, the only possible merges are the pair (b, s_1) if $b < s_1$, or (p_{n_p}, b) if $p_{n_b} > b$. Clearly, when the algorithm stops, either all factors u were merged together among with $p_1 \cdots p_{n_p} b$, or none of the factors u have been merged.

Now, consider the case $a > b$. In such case, it impossible that any of the factors u is merged. By contradiction, suppose it is the case. The first factor l_1 of the Lyndon factorization of z would be of the form $u^k pbs'$ for some suffix s'. But $pbs' < u^k pbs'$ which contradicts the definition of a Lyndon word. It must be that $\mathrm{FLF}(w) = (u, k)$.

In the other case, that is $a < b$, again by contradiction, suppose the above algorithm stops and none of the factors u were merged. In such case, the Lyndon factorization of z is

$$(\underbrace{u, \ldots, u}_{k \text{ times}}, p_1, p_2, \ldots, p_i, p'bs', s_j, \ldots, s_n)$$

where $0 \geq i \geq n_p$, $0 \geq j \geq n_z$, $p' = p_{i+1} p_{i+2} \cdots p_{n_p}$ and $s' = s_1 s_2 \cdots s_{j-1}$. Let α be the word such that $w = pa\alpha$. Since u is a Lyndon word, $u < p'a\alpha$, but $u \geq p'bs' > p'a\alpha$. Contradiction. □

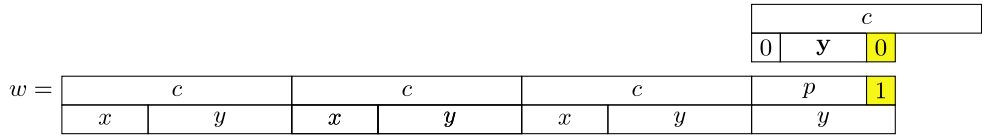
A similar but more precise result is needed in the case of Christoffel words. In order to lighten the presentation, in this section, we only consider words written on the two letter alphabet $\mathcal{A} = \{0 < 1\}$.

Lemma 40. *Given a word w of the form $w = c^k pa$ where $k \geq 1$, $c = (x, y)$ is a non-trivial Christoffel word, $p \in \mathrm{Pref}_+(c)$, $a \in \mathcal{A}$ and $pa \notin \mathrm{Pref}(c)$, then given $z = ws$ for any suffix $s \in \mathcal{A}^*$:*

(1) *if* $|pa| = |y|$ *then* $a = 1$, $pa = y$ *and* $w = c^k pa = (c, c^{k-1}y) \in \mathbf{C}$,
(2) *if* $|pa| \neq |y|$ *and* $a = 0$ *then* $\mathrm{FLF}(z) = (c, k)$,
(3) *if* $|pa| \neq |y|$ *and* $a = 1$ *then* $\mathrm{FLF}_{one}(w) \notin \mathbf{C}$ *and given* $(u, m) = \mathrm{FLF}(f(z))$ *then* $u \in \mathbf{C}$ *and* $|u^m| \geq |w|/2$.

Proof. Through this proof, we use the following notation: given a non-trivial Christoffel word X, its central is noted $\mathbf{X}$, so that $X = 0\mathbf{X}1$, and its complementary Christoffel word is $\hat{X} = 0\overline{\mathbf{X}}1$. Similarly, we also write $p = 0\mathbf{p}$.

(1) By Property 35(c), $p0 \in \mathrm{Pref}(c)$ so that $pa = y$ and



$$c^k pa = (xy)^k y = G^{k-1} \circ D(x, y) = (xy, (xy)^{k-1}y).$$

By Theorem 33, $c^k pa$ is a valid Christoffel word.
(2) Since all Christoffel words are Lyndon words, Lemma 39 applies proving the result.
(3) Again, Lemma 39 applies so $l_1 = \mathrm{FLF}_{one}(w)$ is such that $|l_1| \geq |c^k pa|$.

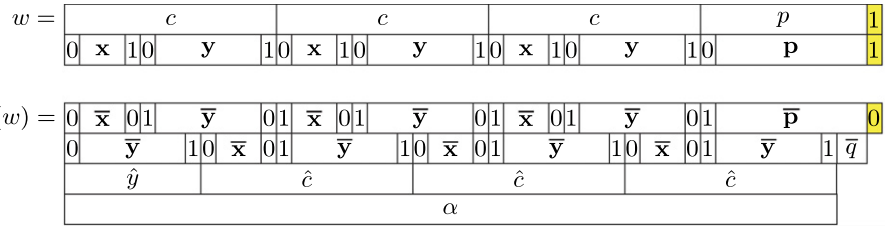
On the other hand, Corollary 34 implies that if $l_1 \in \mathbf{C}$ then it must admit a prefix of the form $c^n y$ with $n \geq k$. Note that y cannot be a trivial Christoffel word since in such case $|p| = 0$ implying $a = 0$. Since $0\mathbf{y}$ is a prefix of c while $y = 0\mathbf{y}1$ is not, $|pa| \neq |y|$ and $|pa| < |c|$ implies that l_1 is not a Christoffel word.

It remains to show that $(u, m) = \mathrm{FLF}(f(w))$ is such that $u \in C$ and $|u^m| \geq |w|/2$.

In a first time, we consider the case where one of the word x is the trivial Christoffel word 0. One checks that in such case $y = 0^n 1$ for some $n \geq 1$, $\mathrm{FLF}(f(wz)) = 01^n(01^{n+1})^{k-1}$ proving the result.

We now assume that both words x and y are non-trivial Christoffel words: $x = 0\mathbf{x}1$ and $y = 0\mathbf{y}1$.

Case 1, $k > 1$:



By Property 35(b), the central words of Christoffel words are palindromes, so that

$$c = 0\mathbf{x}10\mathbf{y}1 = 0\mathbf{y}01\mathbf{x}1 \quad \text{and} \quad \hat{c} = 0\overline{\mathbf{x}}01\overline{\mathbf{y}}1 = 0\overline{\mathbf{y}}10\overline{\mathbf{x}}1.$$

Using this property, we have $f(w) = \hat{y}(\hat{y}\hat{x})^{k-1}\overline{p}$. Moreover, by the unicity of the standard factorization of Christoffel words and Property 35(a), we have

$$c = (x, y) \in \mathbf{C} \iff \hat{c} = (\hat{y}, \hat{x}) \in \mathbf{C}.$$

There are two cases to consider.

1. If $|p| < |y|$.
   One checks that $\alpha = \hat{y}(\hat{y}\hat{x})^{k-1} = \left(\hat{y}(\hat{y}\hat{x})^{k-2}, \hat{y}\hat{x}\right) \in \mathbf{C} \subset \mathbf{L}$ and $0\overline{\mathbf{x}}0\overline{p} \in \mathrm{Pref}_+(\alpha)$ while $0\mathbf{x}0\overline{p}0 \notin \mathrm{Pref}(\alpha)$.
2. If $|p| \geq |y|$.
   In such case, let q be the suffix of p of length $|p| - |y|$. Since p is prefix of $c = 0\mathbf{y}01\mathbf{x}1$, we have that $\overline{\mathbf{p}} = 1\overline{\mathbf{y}}1\overline{q}$. Let $\alpha = \hat{y}\hat{c}^k \in \mathbf{C}$, one checks that $f(w)$ can be written as $f(w) = \alpha\overline{q}0$ where $\overline{q} \in \mathrm{Pref}_+(\alpha)$ and $\overline{q}0 \notin \mathrm{Pref}(\alpha)$.

In both cases, Lemma 39 applies and we have $\mathrm{FLF}(f(z)) = (\alpha, 1)$ with $|\alpha| > |w|/2$.

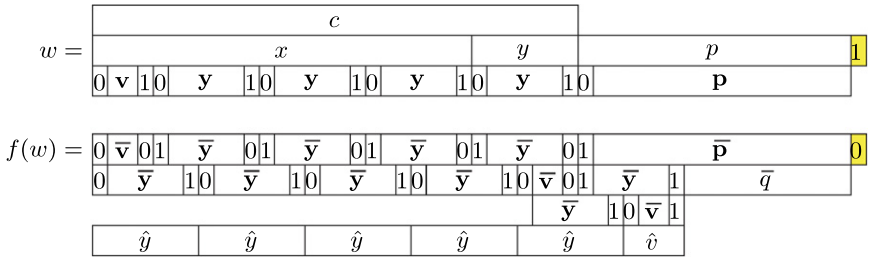
Case 2, $k = 1$.

We assume that $|x| > |y|$, the case where $|x| < |y|$ is shown in a similar way. Note that $c = (x, y)$ forces that $|x| \neq |y|$.

In such case, Theorem 33 implies that there exist a Christoffel word v, $|v| < |y|$, such that $x = vy^n = (vy^{n-1}, y)$ for some $n \geq 1$. Thus w can be written as $w = vy^{n+1}p1$.

In the case where v is the trivial Christoffel word 0 then $y = 0^m 1$ for some $m \geq 1$ and one checks that $\mathrm{FLF}(f(ws)) = (01^m)^{n+1}$ proving the result.

Now consider the case where $v = 0\mathbf{v}1$.



The fact that $\mathbf{c}$, $\mathbf{y}$ and $\mathbf{v}$ are palindromes implies that

$$\bar{\mathbf{c}} = \bar{\mathbf{v}}(01\bar{\mathbf{y}})^{n+1} = (\bar{\mathbf{y}}10)^{n+1}\bar{\mathbf{v}}.$$

Now, depending on the length of p, there are three cases to consider.

i. If $|p| + |v| < |y|$.
We have $f(w) = \hat{y}^{n+1}0\bar{\mathbf{v}}0\bar{p}$ where $\hat{y} \in \mathbf{C} \subset \mathbf{L}$, $0\bar{\mathbf{v}}0\bar{p} \in \mathrm{Pref}_+(\hat{y})$ and $0\bar{\mathbf{v}}0\bar{p}0 \notin \mathrm{Pref}(\hat{y})$. Lemma 39 applies and $\mathrm{FLF}(f(z)) = (\hat{y}, n+1)$ where $|\hat{y}^{n+1}| \geq |w|/2$.

ii. If $|y| \leq |p| + |v| < |y| + |v|$.
In such case, we have that $\bar{\mathbf{p}}$ is a prefix of $\bar{\mathbf{y}}$. Moreover, p is long enough so that $\hat{y}$ is a prefix of $0\bar{\mathbf{v}}01\bar{\mathbf{p}}$ In such case, let q be such that $0\bar{\mathbf{v}}01\bar{\mathbf{p}} = \hat{y}\bar{q}$. Consequently, $f(w)$ may be written as $f(w) = \hat{y}^{n+2}\bar{q}0$ where $\bar{q}$ is a proper prefix of $\hat{y}$ but $\bar{q}0 \notin \mathrm{Pref}(\hat{y})$. We conclude that $\mathrm{FLF}(f(z)) = (\hat{y}, n+2)$ and $|\hat{y}^{n+2}| > |w|/2$.

iii. If $|p| \geq |y|$.
In this case, let q be such that $\bar{p} = 1\bar{\mathbf{y}}1\bar{q}$. Again the palindromic structure of central words allows us to write $\bar{\mathbf{v}}01\bar{\mathbf{y}} = \bar{\mathbf{y}}10\bar{\mathbf{v}}$ so that

$$f(w) = \hat{y}^{n+1}0\bar{\mathbf{v}}01\bar{\mathbf{y}}1\bar{q}0 = \hat{y}^{n+1}0\bar{\mathbf{y}}10\bar{\mathbf{v}}1\bar{q}0 = \hat{y}^{n+2}\hat{v}\bar{q}0.$$

One checks that $\alpha = \hat{y}^{n+2}\hat{v} \in \mathbf{C} \subset \mathbf{L}$ and $\bar{q} \in \mathrm{Pref}_+(\alpha)$ while $\bar{q}0 \notin \mathrm{Pref}(\alpha)$. By Lemma 39, $\mathrm{FLF}(f(z)) = (\alpha, 1)$ with $|\alpha| > |w|/2$. □

References

[1] Jean Berstel, Aldo de Luca, Sturmian words, Lyndon words and trees, Theoret. Comput. Sci. 178 (1–2) (1997) 171–203.
[2] Jean Berstel, Aaron Lauve, Christophe Reutenauer, Franco V. Saliola, Christoffel words and repetitions in words, in: Combinatorics on Words, in: CRM Monograph Series, vol. 27, American Mathematical Society, Providence, RI, 2009.
[3] J.-P. Borel, F. Laubie, Quelques mots sur la droite projective réelle, J. Théor. Nombres Bordeaux 5 (1) (1993) 23–51.
[4] S. Brlek, J.-O. Lachaud, X. Provençal, C. Reutenauer, Lyndon + Christoffel = digitally convex, in: Selected Papers from the 14th IAPR International Conference on Discrete Geometry for Computer Imagery 2008, Pattern Recognit. 42 (10) (2009) 2239–2246.
[5] E.B. Christoffel, Observatio arithmetica, Machine Graphics & Vision 6 (1875) 145–152.
[6] D. Coeurjolly, R. Klette, A comparative evaluation of length estimators of digital curves, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2) (2004) 252–258.
[7] I. Debled-Rennesson, J.-P. Reveillès, A linear algorithm for segmentation of discrete curves, Int. J. Pattern Recognit. Artif. Intell. 9 (1995) 635–662.
[8] Aldo de Luca, Filippo Mignosi, Some combinatorial properties of Sturmian words, Theoret. Comput. Sci. 136 (2) (1994) 361–385.
[9] F. de Vieilleville, J.-O. Lachaud, Digital deformable models simulating active contours, in: S. Brlek, C. Reutenauer, X. Provençal (Eds.), Proc. Int. Conf. Discrete Geometry for Computer Imagery, DGCI'2009, Montréal, Québec, in: Lecture Notes in Computer Science, vol. 5810, Springer, 2009, pp. 203–216.
[10] H. Dörksen-Reiter, I. Debled-Rennesson, Convex and concave parts of digital curves, in: R. Klette, R. Kozera, L. Noakes, J. Weickert (Eds.), Geometric Properties for Incomplete Data, in: Computational Imaging and Vision, vol. 31, Springer, 2006, pp. 145–160.
[11] Jean-Pierre Duval, Factorizing words over an ordered alphabet, J. Algorithms 4 (4) (1983) 363–381.
[12] Fabien Feschet, Laure Tougne, Optimal time computation of the tangent of a discrete curve: application to the curvature, in: Discrete Geometry for Computer Imagery, in: Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 1999, pp. 31–40.
[13] Harold Fredricksen, James Maiorana, Necklaces of beads in k colors and k-ary de Bruijn sequences, Discrete Math. 23 (3) (1978) 207–210.
[14] L.J. Guibas, J. Hershberger, Optimal shortest path queries in a simple polygon, in: SCG'87: Proceedings of the Third Annual Symposium on Computational Geometry, ACM, New York, NY, USA, 1987, pp. 50–63.
[15] J.D. Hobby, Polygonal approximations that minimize the number of inflections, in: SODA'93: Proceedings of the Fourth Annual ACM–SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1993, pp. 93–102.
[16] ImaGene: generic digital image library. Available at: http://gforge.liris.cnrs.fr/projects/imagene.
[17] R. Klette, V. Kovalevsky, B. Yip, On length estimation of digital curves, in: Vision Geometry VIII, vol. 3811, 1999, pp. 117–128.
[18] R. Klette, A. Rosenfeld, Digital Geometry–Geometric Methods for Digital Picture Analysis, Morgan Kaufmann, San Francisco, 2004.
[19] R. Klette, B. Yip, The length of digital curves, Mach. Graph. Vis. 9 (3) (2000) 673–703 (Also Research Report CITR-TR-54, University of Auckland, NZ. 1999).
[20] Jacques-Olivier Lachaud, François de Vieilleville, Convex shapes and convergence speed of discrete tangent estimators, in: George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Paolo Remagnino, Ara Nefian, Gopi Meenakshisundaram, Valerio Pascucci, Jiri Zara, Jose Molineros, Holger Theisel, Thomas Malzbender (Eds.), Advances in Visual Computing, in: Lecture Notes in Computer Science, vol. 4292, Springer, Berlin, Heidelberg, 2006, pp. 688–697.
[21] Jacques-Olivier Lachaud, Anne Vialard, François de Vieilleville, Fast, accurate and convergent tangent estimation on digital contours, Image Vis. Comput. 25 (10) (2007) 1572–1587. Discrete Geometry for Computer Imagery 2005.
[22] M. Lothaire, Combinatorics on Words, in: Cambridge Mathematical Library, Cambridge University Press, Cambridge, 1997.
[23] M. Lothaire, Applied Combinatorics on Words, in: Encyclopedia of Mathematics and its Applications, vol. 105, Cambridge University Press, Cambridge, 2005.
[24] Avraham A. Melkman, On-line construction of the convex hull of a simple polyline, Inform. Process. Lett. 25 (1) (1987) 11–12.

[25] U. Montanari, A note on minimal length polygonal approximation to a digitized contour, Commun. ACM 13 (1) (1970) 41–47.
[26] X. Provençal, J.-O. Lachaud, Two linear-time algorithms for computing the minimum length polygon of a digital contour, in: S. Brlek, C. Reutenauer, X. Provençal (Eds.), Proc. Int. Conf. Discrete Geometry for Computer Imagery, DGCI'2009, Montréal, Québec, in: Lecture Notes in Computer Science, vol. 5810, Springer, 2009, pp. 104–117.
[27] J.-P. Reveillès, Géométrie discrète, calcul en nombres entiers et algorithmique, Thèse d'Etat, Université Louis Pasteur, Strasbourg, France, 1991 (in French).
[28] T. Roussillon, I. Sivignon, L. Tougne, What does digital straightness tell about digital convexity? in: Proc. Int. Workshop on Combinatorial Image Analysis, IWCIA'2009, Mexico, in: Lecture Notes in Computer Science, vol. 5852, Springer, 2009, pp. 43–55.
[29] Luis A. Santaló, Integral Geometry and Geometric Probability, Addison-Wesley Publishing Co., Reading, MA, London, Amsterdam, 1976, With a foreword by Mark Kac, Encyclopedia of Mathematics and its Applications, vol. 1.
[30] J. Sklansky, R.L. Chazin, B.J. Hansen, Minimum perimeter polygons of digitized silhouettes, IEEE Trans. Comput. 21 (3) (1972) 260–268.
[31] F. Sloboda, J. Stoer, On piecewise linear approximation of planar Jordan curves, J. Comput. Appl. Math. 55 (3) (1994) 369–383.
[32] F. Sloboda, B. Zaťko, On one-dimensional grid continua in R^2, Technical Report, Institute of Control Theory and Robotics, Slovak Academy of Sciences, Bratislava, Slovakia, 1996.
[33] F. Sloboda, B. Zaťko, On approximation of Jordan surfaces in 3D, in: G. Bertrand, et al. (Eds.), Digital and Image Geometry, in: LNCS, vol. 2243, Springer, 2001, pp. 365–386.
[34] F. Sloboda, B. Zavťko, J. Stoer, On approximation of planar one-dimensional continua, in: R. Klette, A. Rosenfeld, F. Sloboda (Eds.), Advances in Digital and Computational Geometry, 1998, pp. 113–160.
[35] Klaus Voss, Discrete Images, Objects, and Functions in $\mathbf{Z}^n$, in: Algorithms and Combinatorics, vol. 11, Springer-Verlag, Berlin, 1993.