An Arithmetical Characterization of the Convex Hull of Digital Straight Segments*

Tristan Roussillon

Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205, F-69622, France
tristan.roussillon@liris.cnrs.fr

Abstract. In this paper, we arithmetically describe the convex hull of a digital straight segment by three recurrence relations. This characterization gives new insights into the combinatorial structure of digital straight segments of arbitrary length and intercept. It also leads to two on-line algorithms that computes a part of the convex hull of a given digital straight segment. They both run in constant space and constant time per vertex. Due to symmetries, they are enough to reconstruct the whole convex hull. Moreover, these two algorithms provide efficient solutions to the subsegment problem, which consists in computing the minimal parameters of a segment of a digital straight line of known parameters.

1 Introduction

The connection between continued fractions and the convex hull of lattice points lying above and below a straight segment whose endpoints are lattice points was already observed by Klein in the nineteenth century as mentioned in [7].

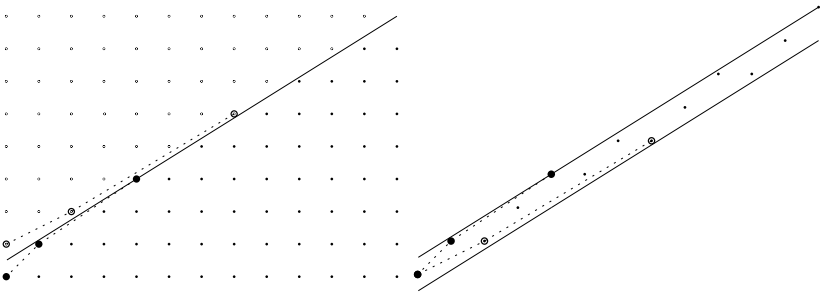
Based on this connection, many papers introduce output-sensitive algorithms to compute the convex hull of analytical point sets, such as the intersection of the fundamental lattice and an arbitrary half-plane [3,6,9,10], convex quadrics [3] or convex bodies [9]. In these papers, the authors propose a *geometrical* extension of the result of Klein, while in this paper, the connection between arithmetic and discrete ray casting, which is briefly described by Har-Peled in [9], is used to propose an arithmetical interpretation of the geometrical algorithm of Charrier and Buzer [6]. This new point of view leads to a simple *arithmetical* extension of the result of Klein to straight segments of arbitrary rational slope and arbitrary rational intercept.

More precisely, we introduce three recurrence relations, defining three sequences of integer pairs, viewed as points or vectors in the fundamental lattice $\mathbb{Z}^2$. The first two sequences, denoted by $\{L_k\}_{0\dots n}$ and $\{U_k\}_{0\dots n}$, both contain vertices of the convex hull of some lattice points lying on each side of a straight line (see fig. 1.a). There exists a close link between a separating line and a digital straight line (DSL). We refer the reader that is not familiar with digital

* This work has been mainly funded by DigitalSnow ANR-11-BS02-009 research grants.

E. Barcucci et al. (Eds.): DGCI 2014, LNCS 8668, pp. 150–161, 2014.

straightness to [11] and we use below the arithmetical framework introduced in [8,15]. For each $0 \leq k \leq n$, $L_k - (0,1)$ (resp. $L_k - (-1,1)$) is a vertex of the lower convex hull of the associated naive (resp. standard) DSL. Fig. 1.b is an illustration of the naive case. For the sake of clarity, we focus on the naive case in the rest of the paper. The last sequence, denoted by $\{v_k\}_{0\ldots n}$, has also a simple geometrical interpretation. Indeed, we prove in section 2.3 that for each $0 \leq k \leq n$, $(L_k - U_k)$ and v_k are a pair of unimodular vectors. In other words, v_k is the direction vector of a digital straight segment (DSS) whose first lower and upper leaning points are respectively $L_k - (0,1)$ and U_k.



(a) separating line (b) digital line

Fig. 1. Upper and lower convex hulls of lattice points of positive x-coordinate lying on each side of the straight line $\{(\alpha,\beta) \in \mathbb{R}^2 | 5\alpha - 8\beta = -4\}$ (Point $(0,0)$ is on the bottom left) (a). They are closely related to the upper and lower convex hulls of a naive and 8-connected digital straight segment of slope $5/8$, intercept -4 and first point $(0,0)$ (b).

Our arithmetical characterization goes beyond the scope of convex hull computation, because the convex hull of a DSS provides a substantial part of its combinatorial structure. Let us define a upper (resp. lower) *digital edge* as a DSS whose first and last point are upper (resp. lower) leaning points. The combinatorial structure of a digital edge has been studied since the seventies [4] and is well-known since the early nineties [5,15,18]. However, these works focus on digital edges or DSLs, which are infinite repetitions of digital edges, because the intercept of a DSL has no effect on its shape and can be assumed to be null without any loss of generality.

To the best of our knowledge, there are few works that extend such results to DSSs of arbitrary intercept and length. In [16], Yaacoub and Reveillès provide an algorithm to retrieve the convex hull of a naive DSS of slope a/b, intercept $\mu \in [0; b[$ and length $|b|$. But the presented algorithm fails to reach the claimed logarithmic complexity, because it takes as input the set of additive convergents of the continued fraction expansion of a/b.

Moreover, it is known for a long time that computing the convex hull of a DSS is a way of computing its parameters [2]. Several authors have recently investigated the problem of computing the minimal parameters of a subsegment of a DSL of known parameters [6,12,14,17]. Minimality is required to have a unique and compact representation of the segment slope and intercept. Due to the prior knowledge of one of its bounding digital straight line, all proposed algorithms outperform classical recognition algorithms [8,13], where each point must be considered at least once. Our simple arithmetical characterization leads to two algorithms, called **smartCH** and **reversedSmartCH**, which not only retrieve the vertices of a part of the DSS convex hull, but also compute its minimal parameters. They both runs in constant space and constant time per vertex. Their overall time complexity are among the best ones (see tab. 1 for a comparison).

Table 1. Theoretical comparison of **smartCH** and **reversedSmartCH** with convex hull algorithms (upper block) and subsegment algorithms (lower block). We consider a naive DSS Σ starting from $(0,0)$ and of slope $a/b = [u_1, \dots, u_n]$ such that $0 \leq a < b$. For the sake of clarity, we consider its left subsegment Σ' of slope $a'/b' = [u'_1, \dots, u'_n]$ and of length $l \leq b$ such that $\Sigma' = \{(x,y) \in \Sigma | 0 \leq x \leq l\}$. Time complexities depend on a, b, and l. If $l \ll b$, bounds depending on l are better. However, if l is close to b, bounds depending on the difference $b - l$ are better.

Algorithms	Time complexity	Remarks
smartCH	$O(\log l)$	on-line: $O(1)$ per vertex
reversedSmartCH	$O(\log(b-l))$	on-line: $O(1)$ per vertex, leaning points must be known
Reveillès et. al. [16]	$O(\sum_i^n u_i)$	$l = b$, $\{u_i\}$ must be known
Har-Peled [9]	$O(\log^2 l)$	on-line: $O(\log l)$ per vertex
Harvey [10]	$O(\log b)$	
Balza-Gomez et. al. [3]	$O(\log l)$	post-processing required
Charrier et. al. [6]	$O(\log l)$	on-line: $O(1)$ per vertex
smartDSS, Lachaud et. al. [12]	$O(\sum_i^n u'_i)$	
reversedSmartDSS, ibid.	$O(\log(b-l))$	$\{u_i\}$ must be known
Sivignon [17]	$O(\log l)$	
Ouattara et. al. [14]	$O(\log l)$	

In section 2, we introduce our arithmetical characterization and discuss its theoretical properties. New algorithms are derived in section 3.

2 A Simple Arithmetical Characterization

Let $\mathcal{L}(a,b,\mu)$ (or simply $\mathcal{L}$) be a straight line of equation $\{(\alpha,\beta) \in \mathbb{R}^2 | a\beta - b\alpha = \mu\}$ with $a, b, \mu \in \mathbb{Z}$, $\gcd(a,b) = 1$. Due to symmetries, let us assume w.l.o.g. that $0 \leq a < b$. In addition, due to invariance by integral translation, let us assume w.l.o.g. that $-b < \mu \leq 0$.

Let Λ be the restriction of the fundamental lattice $\mathbb{Z}^2$ to the lattice points of positive x-coordinate, i.e. $\Lambda := \{(x, y) \in \mathbb{Z}^2 | x \geq 0\}$. The straight line $\mathcal{L}$ always divides Λ into a upper domain, $\Lambda^+ := \{(x, y) \in \Lambda | ax - by \leq \mu\}$, and a lower one, $\Lambda^- := \{(x, y) \in \Lambda | ax - by > \mu\}$.

Definition 1 (Left hull (see fig. 1)). *The lower (resp. upper)* left hull *of Λ^+ (resp. Λ^-) is the part of the lower (resp. upper) convex hull located between the vertex of minimal x-coordinate and the vertex the closest to $\mathcal{L}$.*

In this section, we provide a simple arithmetical characterization of the upper and lower left hull of the lower and upper domain.

2.1 Recurrence Relations

Due to the asymmetric definition of Λ^+ (in which there is a large inequality) and Λ^- (in which there is a strict one), we introduce the two following notations: $\forall x \in \mathbb{R}$ (resp. $\forall x \in \mathbb{R} \setminus 0$), $[x]$ (resp. $\lfloor x \rfloor$) returns the integer $i \in \mathbb{Z}$ farthest to 0 such that $|i| \leq |x|$ (resp. $|i| < |x|$), i and x having same sign. We assume in this paper that these two floor functions run in $O(1)$. Moreover, the restriction of the *strict* floor function $\lfloor \cdot \rfloor$ to $\mathbb{R} \setminus 0$ does not cause any problem in our framework.

On the other hand, we recall that the *remainder* with respect to the straight line of slope a/b is a function $r_{(b,a)} : \mathbb{Z}^2 \to \mathbb{Z}$ such that $r_{(b,a)}(x, y) := (b, a) \wedge (x, y) = ax - by$. This value corresponds to the z-component of the cross-product $(b, a) \wedge (x, y)$ and is equal to the signed area of a parallelogram generated by (b, a) and (x, y). Note that the $\wedge$ operator is linear and antisymmetric.

In the sequel, $r_{(b,a)}(\cdot)$ is simplified into $r(\cdot)$ when the remainder refers to $\mathcal{L}$. Since a and b are given and constant, the difference $r(Q) - \mu$ of a point Q measures how far Q is from $\mathcal{L}$.

Let us consider the following set of recurrence relations (see fig. 2.1 for a numerical example):

$$L_0 = (0, 1),\ U_0 = (0, 0),\ v_0 = (1, 0) + \left[\frac{\mu - a}{-b}\right](0, 1) \tag{1}$$

$$\forall k \geq 1,\ r(v_{k-1}) \neq 0 \quad \begin{cases} \text{if } r(v_{k-1}) > 0, & \begin{cases} L_k = L_{k-1} + \left\lfloor \frac{\mu - r(L_{k-1})}{r(v_{k-1})} \right\rfloor v_{k-1} \\ U_k = U_{k-1} \\ v_k = v_{k-1} + \left[\frac{\mu - (r(U_k) + r(v_{k-1}))}{(r(L_k) - r(U_k))} \right] (L_k - U_k) \end{cases} \\ \text{if } r(v_{k-1}) < 0, & \begin{cases} L_k = L_{k-1} \\ U_k = U_{k-1} + \left[\frac{\mu - r(U_{k-1})}{r(v_{k-1})} \right] v_{k-1} \\ v_k = v_{k-1} + \left\lfloor \frac{\mu - (r(L_k) + r(v_{k-1}))}{(r(U_k) - r(L_k))} \right\rfloor (U_k - L_k) \end{cases} \end{cases} \tag{2}$$

The goal of this section is to prove the following theorem:

Theorem 1. *The sequence $\{L_k\}_{0 \ldots n}$ (resp. $\{U_k\}_{0 \ldots n}$) corresponds to the vertices of the lower (resp. upper) left hull of Λ^+ (resp. Λ^-).*

k	0	1	2	3	4
L_k	$(0,1)$	$(0,1)$	$(2,2)$	$(2,2)$	$(7,5)$
U_k	$(0,0)$	$(1,1)$	$(1,1)$	$(4,3)$	$(4,3)$
v_k	$(1,1)$	$(2,1)$	$(3,2)$	$(5,3)$	$(8,5)$

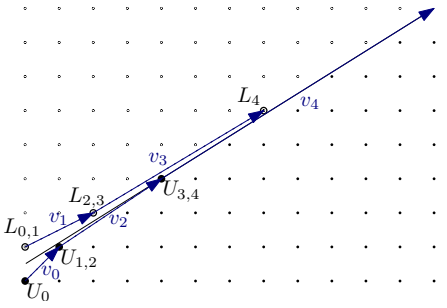


Fig. 2. We apply (1) and (2) for $a = 5$, $b = 8$ and $\mu = -4$. The first two sequences are respectively depicted with white and black disks, whereas the third sequence is depicted with arrows.

The proof of theorem 1 will be derived in section 3 from properties proved in section 2.3. The proof of some of these properties requires the following useful geometrical interpretation of integer divisions (see also [9]).

2.2 Integer Division and Ray Casting

Let us consider a point Q such that $r(Q) \geq \mu$ and a direction vector v whose coordinates are relatively prime and such that $r(v) < 0$.[1] Since $r(Q) \geq \mu$ and $r(v) < 0$, the ray emanating from Q in direction v intersects $\mathcal{L}$ at a point I (see fig. 3.a). The discrete ray casting procedure consists in computing the lattice point farthest from Q and lying on the line segment $[QI]$. Let I be equal to $Q + \tau v$ for some $\tau \in \mathbb{R}^+$. Since I belongs to $\mathcal{L}$ by definition, $r(I) = \mu$ and the linearity of the $\wedge$ product gives:

$$\mu = r(I) = r(Q) + \tau r(v) \Leftrightarrow \tau = \frac{\mu - r(Q)}{r(v)}.$$

Since the components of v are relatively prime and since τ is positive, the greatest integer $t \in \mathbb{Z}$ that is less than or equal to τ, i.e. $t = \lfloor \tau \rfloor$, leads to the lattice point $Q + tv$, which is the farthest from Q among those lying on $[QI]$. In the example illustrated by fig. 3.a, $t = 2$. Note that if we consider the half-open line segment $[QI[$ instead of the closed line segment $[QI]$, i.e. I is not included, we must use the strict floor function $\lfloor \cdot \rfloor$ instead of the large floor function $\lfloor \cdot \rfloor$.

Moreover, note that we can reverse a ray casting under some conditions. We will use this property to propose a dual characterization that leads to a reversed algorithm in section 3. Let us recall that the *position* of a point with respect to a direction vector s is a function $p_s : \mathbb{Z}^2 \to \mathbb{Z}$ such that $p_s(x, y) := (x, y) \wedge s$. In the sequel, we assume that $s = (0, 1)$ because we focus on the naive case and $p_s(x, y)$, which is merely denoted by $p(x, y)$, returns x.

[1] Note that points and vectors are both viewed as integer pairs $(x, y) \in \mathbb{Z}^2$.

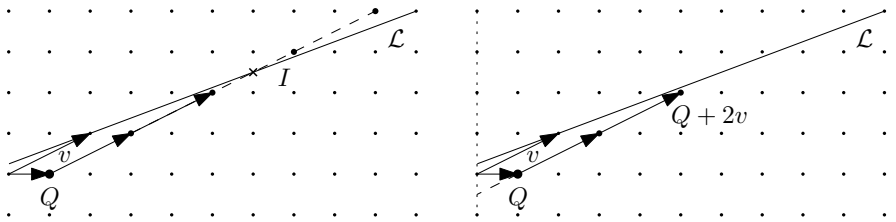


Fig. 3. In (a), the ray emanating from Q in direction v intersects $\mathcal{L} = \{(\alpha, \beta) \in \mathbb{R}^2 | 3\alpha - 8\beta = -2\}$ at I, because $Q = (1, 0)$ and $v = (2, 1)$ are such that $r(Q) = 3 \geq \mu = -2$ and $r(v) = -2 < 0$. The lattice point lying on the ray segment $[QI]$ and farthest from Q is $Q + 2v = (5, 2)$. Indeed, $[\frac{\mu - r(Q)}{r(v)}] = [\frac{5}{2}] = 2$. In (b), the point Q may be retrieved from $Q + 2v$ and v by a reversed discrete ray casting procedure, because $p(Q) = 1$ is strictly less than $p(v) = 2$.

It is easy to see that

$$p(Q) < p(v) \Rightarrow \left[\frac{p(Q + tv)}{p(v)}\right] = t. \tag{3}$$

Fig. 3.b shows the reversed version of the ray casting depicted in fig. 3.a.

2.3 A Unimodular Basis

We now prove several properties of (1) and (2). Let n be the index such that $r(v_n) = 0$. For the sake of clarity, we postpone the demonstration of the existence of such index to the end of the subsection. We first show that for each $0 \leq k \leq n$, points L_k and U_k lie on each side of $\mathcal{L}$, i.e.

$$\forall 0 \leq k \leq n,\ r(L_k) < \mu \text{ and } r(U_k) \geq \mu. \tag{4}$$

It is easy to see that (4) is true for $k = 0$ by (1) and that for all $0 \leq k \leq n$, the constructions of L_k from L_{k-1} (when $r(v_{k-1}) > 0$) and U_k from U_{k-1} (when $r(v_{k-1}) < 0$) are such that $r(L_k) < \mu$ and $r(U_k) \geq \mu$.

Moreover, for each $0 \leq k \leq n$, we show that there is a strong link between L_k, U_k and v_k:

Lemma 1. $\forall 0 \leq k \leq n$, *v_k is the unique negative and* valid Bezout vector[2] *of* $(L_k - U_k)$, *i.e.*

$$\forall 0 \leq k \leq n,\ r(L_k) < \mu - r(v_k) \leq r(U_k), \tag{5}$$

$$\forall 0 \leq k \leq n,\ v_k \wedge (L_k - U_k) = -1. \tag{6}$$

[2] The notion of valid Bezout vector is introduced in [6].

To prove lemma 1, we prove successively (5) and (6).

Proof (of (5)). Base case: Let us consider the ray emanating from $(1,0)$ in direction $L_0 = (0,1)$. The lattice point farthest from $(1,0)$ lying on the ray and below $\mathcal{L}$ is v_0 by (1). We have thus $r(v_0) \geq \mu$ and as a corollary, $r(v_0) + r(L_0) < \mu$, because the lattice point following v_0 in direction L_0 is above $\mathcal{L}$. Putting the two inequalities together we have $r(L_0) < \mu - r(v_0) \leq r(U_0)$.

Induction step: Let us assume that for some k between 1 and n, $r(L_{k-1}) < \mu - r(v_{k-1}) \leq r(U_{k-1})$. We focus on the case where $r(v_{k-1}) > 0$, because the other case is symmetric. In that case, due to the induction hypothesis, $r(U_k) = r(U_{k-1}) \geq \mu - r(v_{k-1})$. Let us consider now the ray emanating from $U_k + v_{k-1}$ in direction $L_k - U_k$. It intersects $\mathcal{L}$ because (i) $r(U_k + v_{k-1}) \geq \mu$ and (ii) $r(L_k - U_k) < 0$ by (4). The lattice point farthest from $U_k + v_{k-1}$ lying on the ray and below $\mathcal{L}$ is $U_k + v_k$ by (2). We have thus $r(U_k) + r(v_k) \geq \mu$ and as a corollary, $r(U_k) + r(v_k) + r(L_k - U_k) < \mu$, which is equivalent to $r(L_k) + r(v_k) < \mu$. As a consequence, we have $r(L_k) < \mu - r(v_k) \leq r(U_k)$, which concludes the proof. □

To show (6), we use induction and the properties of the $\wedge$ operator (linearity and anticommutativity).

Proof (of (6)). Base case: $v_0 = (1,0) - c(0,1)$ for some constant $c \in \mathbb{Z}$ and $(L_0 - U_0) = (0,1)$ by (1). Therefore $v_0 \wedge (L_0 - U_0) = (1,0) \wedge (0,1) = -1$.

Induction step: let us assume that for some k between 1 and n, $v_{k-1} \wedge (L_{k-1} - U_{k-1}) = -1$. By (2), $v_k = v_{k-1} - c(L_k - U_k)$ and $(L_k - U_k) = (L_{k-1} - U_{k-1}) - c' v_{k-1}$ for some constants $c, c' \in \mathbb{Z}$. We conclude that $v_k \wedge (L_k - U_k) = v_{k-1} \wedge (L_{k-1} - U_{k-1})$, which is equal to -1 due to the induction hypothesis. □

Note that (6) implies that $\forall 0 \leq k \leq n$, v_k and $(L_k - U_k)$ are *irreducible*, i.e. their coordinates are relatively prime. Indeed, v_0 and $(L_0 - U_0)$ are irreducible and for all $k \geq 1$, the greatest common divisor of their coordinates divides $v_k \wedge (L_k - U_k)$, which is equal to -1.

Geometrically, (6) implies that at each step $0 \leq k \leq n$, v_k and $(L_k - U_k)$ are a pair of unimodular vectors, while (4) and (5) guarantee that in such a basis the line segment $[L_k U_k]$ is always intersected by $\mathcal{L}$, whereas the line segment of endpoints L_k (resp. U_k) and $L_k + v_k$ (resp. $U_k + v_k$) is never intersected by $\mathcal{L}$.

We now end the subsection with the following lemma:

Lemma 2. *There exists a step $n \geq 1$ such that:*

$$r(L_n - U_n) = -1,\ r(v_n) = 0. \tag{7}$$

$$U_n = \mu \text{ and } L_n = \mu - 1. \tag{8}$$

To prove lemma 2, it is enough to notice that $r(L_k - U_k)$ is always strictly negative by (4) and that $\forall 0 \leq k \leq n$, $r(L_{k-1} - U_{k-1}) < r(L_k - U_k)$ by (2). These inequalities and (6) guarantee that $r(L_n - U_n) = -1$ and $r(v_n) = 0$ at some step $n \geq 1$. Note that $v_n = (b, a)$, because v_n is irreducible. Then, by (4), the only possible values of $r(L_n)$ and $r(U_n)$ must be respectively $\mu - 1$ and μ.

3 Convex Hull Algorithms

The proof of theorem 1 is now straightforward:

- Ray casting is equivalent to integer division on remainders (section 2.2),
- (1) is equivalent to the initialization of Charrier and Buzer's algorithm [6].
- By lemma 1, (1) and (2) maintain the same invariant as Charrier and Buzer's algorithm [6], i.e. $\forall 0 \leq k \leq n$, v_k is the negative and valid bezout vector of $(L_k - U_k)$.
- Lemma 2 guarantees that the whole lower and upper left hulls are computed.

As a consequence, (1) and (2) provide a simple way to compute the left hull of the lower and upper domains. In the following sections, we show how to translate (1) and (2) first into a forward algorithm and then into a backward algorithm, based on (3).

3.1 A Forward Approach

We first propose a forward algorithm that computes L_k, U_k and v_k with increasing k, starting from L_0, U_0 and v_0. This algorithm is called **smartCH**, because it is on-line and runs in $O(1)$ per vertex and is thus optimal (see algorithm 1).

It is a rather straightforward translation of (1) and (2). There is a difference though: we add an extra constraint, which modifies the stopping criterion. Algorithm **smartCH** takes as input not only the slope a/b and the intercept μ of a DSL, but also the length l of the subsegment Σ' starting from $(0, 0)$. If b is minimal for Σ' or, which is equivalent, if $l \gg b$, the algorithm iteratively computes L_k, U_k, and v_k from L_{k-1}, U_{k-1} and v_{k-1} until $r(v_k) = 0$. Otherwise, the algorithm stops as soon as it detects that a new lower or upper leaning point would lie outside Σ' (lines 3 and 7 of algorithm 2). In this case, we correct the last ray casting in order to get the last leaning point (lines 14 and 17) or the last direction vector of Σ' (lines 9 to 11 and 18 to 19 of algorithm 2). The components of the last direction vector gives the rational slope whose denominator is bounded by l and that best approximates a/b. This final step is computed in $O(1)$ instead of the $O(\log(l))$ steps required to compute the critical supporting lines of the lower and upper convex hulls as proposed in [6].

3.2 A Backward Approach

If all partial quotients are known, by using (1) and (2), it is obviously possible to compute U_k, L_k, and v_k with decreasing k from a given step $n \geq 1$. In this section, we show that these partial quotients can be computed from the positions of U_n, L_n and v_n.

As seen in section 2.2, ray casting can be reversed under some conditions. These conditions are actually fulfilled in our framework:

Algorithm 1. smartCH(a, b, μ, l)

Input: a, b, μ, l
Output: V, LHull and UHull, lower and upper left hull
`// initialisation`
stop := FALSE ;
U := (0,0) ; add U to UHull ;
L := (0,1) ; add L to LHull ;
V := (1,0) + $\left\lceil \frac{\mu - a}{-b} \right\rceil$ (0,1) ;
`// main loop`
while $r(v_k) \neq 0$ *and not stop* **do**
  if $r(v_k) > 0$ **then**
    stop := nextVertex(a, b, μ, l, L, U, V, LHull, [.], $\lfloor . \rfloor$) ;
  if $r(v_k) < 0$ **then**
    stop := nextVertex(a, b, μ, l, U, L, V, UHull, $\lfloor . \rfloor$, [.]) ;

Algorithm 2. nextVertex$(a, b, \mu, l$, X, Y, V, XHull, $floor_1$, $floor_2)$

Input: a, b, μ, l, X, Y, V, XHull, $floor_1$, $floor_2$
Output: X, V, XHull
q := $floor_1\left(\frac{\mu - r(X)}{r(V)}\right)$; `// first ray casting`
X := X + q V ;
if $(p(X) \leq l)$ **then**
  add X to XHull ;
  q := $floor_2\left(\frac{\mu - (r(Y) + r(V))}{(r(X) - r(Y))}\right)$; `// second ray casting`
  V := V + q (X - Y) ;
  if $(p(Y) + p(V) \leq l)$ **then return** *TRUE* ;
  else
    V := V - q (X - Y) ;
    q := $\left\lceil \frac{l - (p(Y) + p(V))}{p(X) - p(Y)} \right\rceil$;
    if $q > 0$ **then** V := V + q (X - Y) ;
    return *FALSE* ;
else
  X := X - q V ;
  q := $\left\lceil \frac{l - p(X)}{p(V)} \right\rceil$;
  if $q > 0$ **then**
    X := X + q V ; add X to XHull ;
  q := $\left\lceil \frac{l - (p(Y) + p(V))}{p(X) - p(Y)} \right\rceil$;
  if $q > 0$ **then** V := V + q (X - Y) ;
  return *FALSE* ;

Theorem 2. *For each $0 \leq k \leq n$, the positions of L_k, U_k, v_k are ordered as follows:*

$$\forall 0 \leq k < n, \begin{cases} p(v_k) < p(L_{k+1}), & \text{if } r(v_k) > 0 \\ p(v_k) < p(U_{k+1}), & \text{if } r(v_k) < 0. \end{cases} \tag{9}$$

$$\forall 0 \leq k \leq n,\ p(L_k) < p(v_k) \text{ and } p(U_k) < p(v_k). \tag{10}$$

Inequalities (9) are obvious and provide the necessary and sufficient condition to reverse the second ray casting (lines 5-6 of algorithm 2). Indeed, (3) requires that

$$p(Y) + p(V) < p(X - Y) \Leftrightarrow p(V) < p(X)$$

and according to the notation used in algorithm 2, $X = L_{k+1}$ if $V = v_k$ has a positive remainder, but $X = U_{k+1}$ otherwise.

Inequalities (10) provide the necessary and sufficient condition to reverse the first ray casting (lines 1-2 of algorithm 2). Indeed, (3) requires that $p(X) < p(V)$ and according to the notation used in algorithm 2, $X = L_k$ if $V = v_k$ has a positive remainder, but $X = U_k$ otherwise.

To complete the proof of theorem 2, we prove (10) by induction.

Proof (of (10)). Base case: Since $p(v_0) = 1$ while $p(L_0) = p(U_0) = 0$, (10) is obviously true for $k = 0$.

Induction step: Let us assume that $p(L_{k-1}) < p(v_{k-1})$ and that $p(U_{k-1}) < p(v_{k-1})$ for some k between 1 and n. Let us assume that $r(v_{k-1}) > 0$, the other case being symmetric. By (2), it is easy to see that $p(v_k) \geq p(v_{k-1}) + p(L_k - U_k)$. Since $U_k = U_{k-1}$ and $p(v_{k-1}) - p(U_{k-1}) > 0$ due to the induction hypothesis, we have $p(v_k) > p(L_k)$. Since $p(L_k) > p(U_k)$, we obviously have also $p(v_k) > p(v_{k-1}) > p(U_{k-1}) = p(U_k)$. □

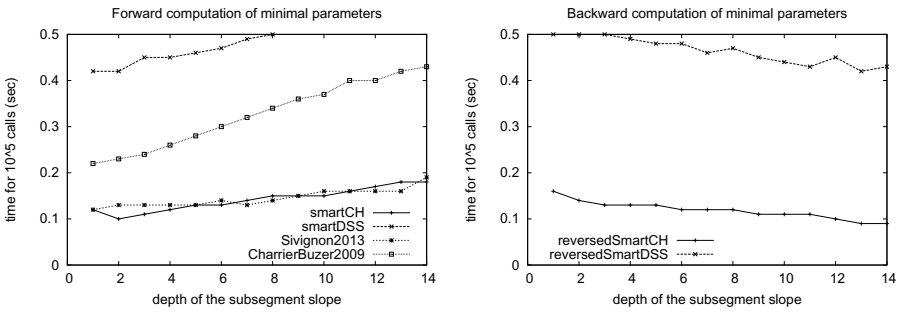
Theorem 2 and (3) lead to a new set of recurrence relations, which is dual to (1) and (2):

$$\forall k \leq n,\ p(U_k) \neq p(L_k) \quad \begin{cases} \text{if } p(L_k) < p(U_k), & \begin{cases} L_{k-1} = L_k + \left\lceil \frac{p(L_k)}{p(v_{k-1})} \right\rceil v_{k-1} \\ U_{k-1} = U_k \\ v_{k-1} = v_k + \left\lfloor \frac{p(v_k) - p(U_k)}{p(U_k) - p(L_k)} \right\rfloor (L_k - U_k) \end{cases} \\ \text{if } p(L_k) > p(U_k), & \begin{cases} L_{k-1} = L_k \\ U_{k-1} = U_k + \left\lceil \frac{p(U_k)}{p(v_{k-1})} \right\rceil v_{k-1} \\ v_{k-1} = v_k + \left\lfloor \frac{p(v_k) - p(L_k)}{p(U_k) - p(L_k)} \right\rfloor (U_k - L_k) \end{cases} \end{cases} \tag{11}$$

This set of recurrence relations has properties similar to (2) and straightforwardly leads to a backward algorithm, called **reversedSmartCH**, which is on-line and runs in $O(1)$ per vertex. As done with **smartCH**, we can add a length constraint to stop the algorithm sooner and solve the subsegment problem. We do not provide more details due to lack of space. However, we compare below our implementation of **smartCH** and **reversedSmartCH** to the algorithms whose implementation is available in DGtal [1].

3.3 Experiments

We generate random DSSs, starting from $(0,0)$, whose slope a/b has a continued fraction expansion of constant depth n equal to 15.[3] For each DSS, we consider one of its left subsegment starting from $(0,0)$. Its length l is determined by the denominator of the k-th convergent of the bounding DSS slope, so that the subsegment slope of minimal denominator has a continued fraction expansion of depth n' equal to k. In fig. 4, we plot the running times (in seconds) of 100.000 calls of several algorithms against parameter k, which is ranging from 1 to 14. As expected in tab. 1, we observe that the running time of the forward methods (a) is linear in n' (and thus logarithmic in l), whereas the running time of the backward methods (b) is linear in $n - n'$ (and thus logarithmic in $b - l$). Our generic implementation of **smartCH** outperforms [6], smartDSS [12] and is comparable to [17]. Moreover, our implementation of **reversedSmartCH** outperforms reversedSmartDSS [12], which is much more space consuming because continued fraction expansions of all DSS slopes are stored in a shared data structure that grows at each call.



(a) (b)

Fig. 4. We plot the running times of the algorithms whose implementation is available in DGtal [1], i.e. [6,12,17], against parameter k, which is equal to the depth of the subsegment slope

4 Conclusion

In this paper, we propose a simple arithmetical characterization of the convex hull of DSSs, which gives new insights into the combinatorial structure of DSSs of arbitrary intercept and length. This characterization and its dual, lead to two on-line algorithms that computes the left hull of a given DSS. The first one, called **smartCH**, returns vertices of decreasing remainders, but increasing positions, while the second one, called **reversedSmartCH**, returns vertices of

[3] Note that the depth is set to 15 and all partial quotients are randomly chosen in $\{1, \ldots, 4\}$ so that numerators and denominators are not greater than $2^{31} - 1$.

increasing remainders, but decreasing positions. They both run in constant space and constant time per vertex. They also provide a logarithmic-time and efficient solution to the subsegment problem.

References

1. DGtal: Digital geometry tools and algorithms library, `http://libdgtal.org`
2. Anderson, T.A., Kim, C.E.: Representation of digital line segments and their preimages. Computer Vision, Graphics, and Image Processing 30(3), 279–288 (1985)
3. Balza-Gomez, H., Moreau, J.-M., Michelucci, D.: Convex hull of grid points below a line or a convex curve. In: Bertrand, G., Couprie, M., Perroton, L. (eds.) DGCI 1999. LNCS, vol. 1568, pp. 361–374. Springer, Heidelberg (1999)
4. Brons, R.: Linguistic Methods for the Description of a Straight Line on a Grid. Computer Graphics and Image Processing 3(1), 48–62 (1974)
5. Bruckstein, A.M.: Self-Similarity Properties of Digitized Straight Lines. Contemporary Mathematics 119, 1–20 (1991)
6. Charrier, E., Buzer, L.: Approximating a real number by a rational number with a limited denominator: A geometric approach. Discrete Applied Mathematics 157(16), 3473–3484 (2009)
7. Davenport, H.: The Higher Arithmetic: Introduction to the Theory of Numbers. Oxford University Press, Oxford (1983)
8. Debled-Rennesson, I., Reveillès, J.P.: A linear algorithm for segmentation of digital curves. International Journal of Pattern Recognition and Artificial Intelligence 9(4), 635–662 (1995)
9. Har-Peled, S.: An output sensitive algorithm for discrete convex hulls. Computational Geometry 10(2), 125–138 (1998)
10. Harvey, W.: Computing Two-Dimensional Integer Hulls. SIAM Journal on Computing 28(6), 2285–2299 (1999)
11. Klette, R., Rosenfeld, A.: Digital straitghness – a review. Discrete Applied Mathematics 139(1-3), 197–230 (2004)
12. Lachaud, J.O., Said, M.: Two efficient algorithms for computing the characteristics of a subsegment of a digital straight line. Discrete Applied Mathematics 161(15), 2293–2315 (2013)
13. Lindenbaum, M., Bruckstein, A.: On recursive, $o(n)$ partitioning of a digitized curve into digital straight segments. IEEE Transactions on Pattern Analysis and Machine Intelligence 15(9), 949–953 (1993)
14. Ouattara, J.S.D., Andres, E., Largeteau-Skapin, G., Zrour, R., Tapsob, T.M.Y.: Remainder Approach for the Computation of Digital Straight Line Subsegment Characteristics. Submitted to Discrete Applied Mathematics (2014), doi:10.1016/j.dam.2014.06.006
15. Reveillès, J.P.: Géométrie Discrète, calculs en nombres entiers et algorithmique. Thèse d'etat, Université Louis Pasteur (1991)
16. Reveillès, J.P., Yaacoub, G.: A sublinear 3D convex hull algorithm for lattices. In: DGCI 1995, pp. 219–230 (1995)
17. Sivignon, I.: Walking in the Farey Fan to Compute the Characteristics of a Discrete Straight Line Subsegment. In: Gonzalez-Diaz, R., Jimenez, M.-J., Medrano, B. (eds.) DGCI 2013. LNCS, vol. 7749, pp. 23–34. Springer, Heidelberg (2013)
18. Voss, K.: Coding of digital straight lines by continued fractions. Computers and Artificial Intelligence 10(1), 75–80 (1991)