Combinatorial Generation of Planar Sets [*]

Tristan Roussillon

December 7, 2023

Abstract

We introduce a multi-dimensional generalization of the Euclidean algorithm and show how it is related to digital geometry and particularly to the generation and recognition of digital planes. We show how to associate with the steps of the algorithm geometrical extensions of substitutions, i.e., rules that replace faces by unions of faces, to build finite sets called patterns. We examine several of their combinatorial, geometrical and topological properties. This work is a first step towards the incremental computation of patterns that locally fit a digital surface for the accurate approximation of tangent planes.

1 Introduction

Digital geometry mainly deals with sets of discrete elements considered to be digitized versions of Euclidean objects. A digital surface may be seen as a mesh of unit square faces whose vertices have integer coordinates. A challenge is to decompose digital surfaces into patches, such as pieces of digital planes.

A digital plane has been analytically defined as a set of points with integer coordinates lying between two parallel planes [23]. The topological properties of such object depend on the thickness of the strip. If one considers the graph whose nodes are points of integer coordinates and whose edges link points which are distant to each other less than a given thickness, one can efficiently generate an arbitrary piece of digital plane using a breadth-first search and the analytical definition as a set-membership predicate. Furthermore, given a finite point set, one can decide whether this set belongs to a digital plane or not in linear time using linear programming. See for instance [13] and, for a review on digital planarity, [7]. However, a linear programming solver does not help so much for the analysis of digital surfaces, because one does not know which point set should be tested to obtain patches that approximate the tangent planes of the underlying surface.

In order to cope with this problem, *plane-probing* algorithms have been developed [19, 20]. Their main feature is to decide on-the-fly how to probe a given point set and locally align a triangle with it, without requiring any user input. However, if probing for points in a sparse way is perfect for digital planes, it is not enough for non-convex parts where the triangles may jump over holes or stab the digital surface. Therefore, that approach also requires to associate pieces of digital planes to the triangles and check whether they fit the digital surface or not.

In this paper, we make a first step towards the incremental generation of planar patches, which can be used during the execution of a plane-probing algorithm (see Fig. 1). To do that, we take advantage of the combinatorial properties of digital planes.

The particular case of digital lines has been studied for a long time in different contexts and has led to many applications. One key result is that digital lines are hierarchical point sets whose structure is exactly described by the continued fraction expansion of their slope and strongly

[*]This work has been funded by PARADIS ANR-18-CE23-0007-01 research grant.

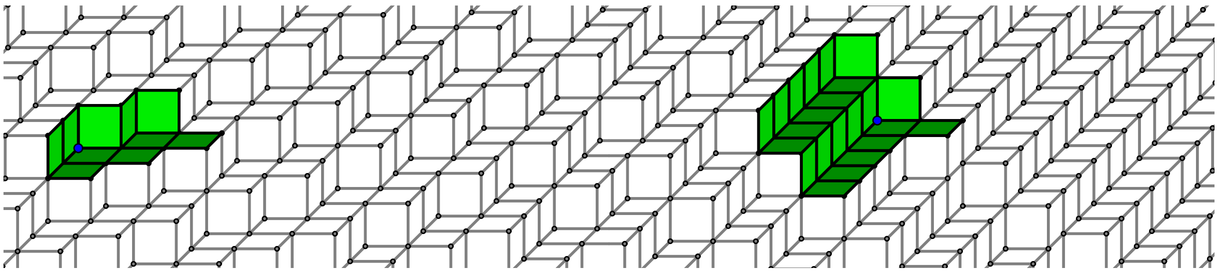


Figure 1: Local approximation of a digital sphere of radius 63 by planar patches (in green) from two starting points (in blue). The implementation combines the plane-probing algorithm H [20] with the generation method of Section 3.

relies on the Euclidean algorithm. See for instance [25] and, for a survey on digital straightness, [16].

There has been much effort done in order to find similar results in three dimensions despite the lack of a canonical algorithm and the diversity of existing generalizations of the Euclidean algorithm. Some combinatorial results, involving symmetries, piece exchanges and flips, have been stated thanks to an appropriate representation of digital planes [18]. However, most of other related works depend on a multi-dimensional generalization of the Euclidean algorithm such as *Brun*, e.g., [4], *Jacobi-Perron*, e.g., [6], *Fully substractive*, e.g. [3], or a mix of several of them [14]. Those algorithms have been used to generate digital planes from a normal vector. There are two different but closely related construction schemes in the literature. The first one is based on union and translation of point sets [5, 10, 14]. It has been used mostly to construct the thinnest digital plane that is connected [3, 5, 8, 9]. The second one is based on a description of standard digital planes as unions of square faces and uses geometrical extensions of substitutions, i.e. rules that replace square faces by unions of square faces. Since the pioneer work of Arnoux and Ito [2], that formalism has been used for instance in [4, 5, 6, 11, 12]. Both construction schemes incrementally generate sets so that the current set will be included in the next one, but suffer from topological and geometrical limitations. This work is based on the second scheme that generates few elements in comparison with the first scheme, e.g., [14].

This paper is an extension of [22] and is organized as follows. We first present a multi-dimensional generalization of the Euclidean algorithm in Section 2. It includes a far larger class of existing algorithms than the generalization proposed in [22]. Furthermore, we do not only show that plane-probing algorithms look like a geometrical version of a three-dimensional Euclidean algorithm, but also show that all the classical three-dimensional Euclidean algorithms like Brun or Jacobi-Perron can be translated into plane-probing algorithms. In Section 3, we explain how one can generate unions of faces from the output of the above-mentioned algorithms. We also prove several combinatorial properties of the generated sets. Several of them are new, others are stronger results or have a more self-contained proof than the ones included in [22].

2 Generalization of the Euclidean Algorithm

We first propose in Subsection 2.1 a multi-dimensional generalization of the Euclidean algorithm: Algorithm 1. Its validity and termination are then established in Subsection 2.2. Algorithm 1 is linked with several other known algorithms in the last two subsections.

2.1 Notations, Definitions and Algorithm

Let $\{\mathbf{e}_1, \dots, \mathbf{e}_d\}$ be the canonical basis of $\mathbb{R}^d$, with $d \geq 2$. We denote $\mathbf{0}$ the origin and $\mathbf{1} = \sum_{i=1}^{d} \mathbf{e}_i$ the vector with all coordinates equal to 1 whatever the dimension d. Let $\mathbf{I}_d$ be the $d \times d$ identity matrix. Matrices and vectors are written in bold, respectively with upper-case and lower-case letters. Vectors are thought as column vectors, while their transpose, in front of which is put the letter t, are thought as row vectors. For instance, the i-th coordinate of $\mathbf{x}$ is equal to ${}^{\mathrm{t}}\mathbf{e}_i\mathbf{x}$. However, for sake of shortness, ${}^{\mathrm{t}}\mathbf{e}_i\mathbf{x}$ will be also denoted x_i.

Let $\mathbb{N}^d$ be the set of vectors of length d with nonnegative integer coordinates. Let $\leq$ be the coordinatewise partial order defined such that, for all pairs $\mathbf{x}, \mathbf{y} \in \mathbb{N}^d$, $\mathbf{x} \leq \mathbf{y}$ if and only if $x_i \leq y_i$ for all $i \in \{1, \ldots, d\}$. Endowed with the order relation $\leq$, $\mathbb{N}^d$ is a *lattice.*

The Euclidean algorithm is often used to compute the greatest common divisor of two numbers, but in what follows we are only interested in the sequence of operations performed when the algorithm is executed. Since multiplying the two numbers by a constant does not change the sequence of operations, we can assume that the greatest common divisor is equal to 1. More generally, for d numbers, we can similarly focus on vectors of $\mathbb{N}^d$ whose coordinates are relatively prime, i.e., on the set $\mathbb{P}^d := \{\mathbf{x} \in \mathbb{N}^d \mid \gcd(x_1, \ldots, x_d) = 1\}$. Note that $\mathbb{P}^d$ contains $\mathbf{e}_1, \ldots, \mathbf{e}_d$ and $\mathbf{1}$, but not $\mathbf{0}$.

We now define the set of inputs accepted in Algorithm 1:

$$\text{for each } S \subset \mathbb{P}^d, \Omega(S) := \bigcup_{\mathbf{x} \in S} \{\mathbf{y} \in \mathbb{P}^d \mid \mathbf{x} \leq \mathbf{y}\}.$$

We will only consider $\Omega(\{\mathbf{1}\})$ and $\Omega(\{\mathbf{e}_1, \ldots, \mathbf{e}_d\})$. In the first set, the coordinates of the vectors are coprime and strictly positive, whereas they are coprime and nonnegative in the second one.

Algorithm 1 takes as input a set $T \subset \mathbb{P}^d$ whose elements are called *terminal* and a vector $\mathbf{v} \in \Omega(T)$. While $\mathbf{v}$ is not terminal, it is iteratively transformed into another vector by matrix multiplication. The matrices must belong to $\mathbb{U}^d$, defined as the set of $d \times d$ matrices having -1 at the intersection between the k-th row and the l-th column with distinct $k, l \in \{1, \ldots, d\}$ and the same entries as $\mathbf{I}_d$ elsewhere. Note that the elements of $\mathbb{U}^d$ are matrices with determinant 1 and are thus unimodular. Note that multiplication by a unimodular matrix $\mathbf{U}$ preserves vectors of coprime coordinates, i.e., $\mathbf{v} \in \mathbb{P}^d \Rightarrow \mathbf{U}\mathbf{v} \in \mathbb{P}^d$. This fact is used in the next section.

Algorithm 1 Multidimensional Euclidean algorithm.

```
Require: a terminal set $T$ and a vector $\mathbf{v} \in \Omega(T)$
Ensure: a list of matrices $(\mathbf{U}_n)_{n \in \{0,\ldots,N\}}$ and a terminal $\mathbf{t} \in T$ such that $\mathbf{t} = \mathbf{U}_N \cdots \mathbf{U}_0 \mathbf{v}$.
1: lst ← $\{\mathbf{I}_d\}$
2: while $\mathbf{v} \notin T$ do
3:     Select $\mathbf{U} \in \mathbb{U}^d$ such that $\mathbf{U}\mathbf{v} \neq \mathbf{v}$ and $\mathbf{U}\mathbf{v} \in \Omega(T)$
4:     lst ← lst $\cup \{\mathbf{U}\}$
5:     $\mathbf{v} \leftarrow \mathbf{U}\mathbf{v}$
6: end while
7: return lst, $\mathbf{v}$
```

2.2 Validity, Termination and Complexity

In this subsection, we answer two questions regarding Algorithm 1: is there always a matrix $\mathbf{U} \in \mathbb{U}^d$ satisfying the constraints on line 3? Does the algorithm always terminate?

Before answering, we introduce the following technical result:

Lemma 1. *Let T be either equal to $\{\mathbf{1}\}$ or $\{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$. Let us consider a vector $\mathbf{v} \in \Omega(T)$ and a matrix $\mathbf{U} \in \mathbb{U}^d$ containing -1 at the intersection of the k-th row and the l-th column, with distinct $k, l \in \{1, \ldots, d\}$.*

The following relations involving the coordinates of $\mathbf{v}$,

- $v_l \geq 1$ *and,*
- *(1)* $v_k - v_l \geq 1$, *if* $T = \{\mathbf{1}\}$,

(2) $v_k - v_l \geq 0$*, if* $T = \{\mathbf{e}_1, \dots, \mathbf{e}_d\}$*,*

imply both $\mathbf{U}\mathbf{v} \neq \mathbf{v}$ *and* $\mathbf{U}\mathbf{v} \in \Omega(T)$*.*

Proof. The k-th coordinate of $\mathbf{U}\mathbf{v}$ is the result of substracting the l-th coordinate from the k-th coordinate of $\mathbf{v}$, i.e., ${}^{\mathrm{t}}\mathbf{e}_k(\mathbf{U}\mathbf{v}) = v_k - v_l$. The other coordinates of $\mathbf{U}\mathbf{v}$ are the same as $\mathbf{v}$.

On one hand, we have $\mathbf{U}\mathbf{v} \neq \mathbf{v}$ if $v_l \geq 1$. On the other hand, we have $\mathbf{U}\mathbf{v} \in \Omega(T)$ if and only if $\mathbf{U}\mathbf{v} \in \mathbb{P}^d$ and there exists $\mathbf{t} \in T$ such that $\mathbf{t} \leq \mathbf{U}\mathbf{v}$. The first condition is true because $\mathbf{v} \in \mathbb{P}^d$ and $\mathbf{U}$ is unimodular. The second condition is true if ${}^{\mathrm{t}}\mathbf{e}_k(\mathbf{U}\mathbf{v}) \geq 1$ (resp. ${}^{\mathrm{t}}\mathbf{e}_k(\mathbf{U}\mathbf{v}) \geq 0$) if $T = \{\mathbf{1}\}$ (resp. $T = \{\mathbf{e}_1, \dots, \mathbf{e}_d\}$).

Since ${}^{\mathrm{t}}\mathbf{e}_k(\mathbf{U}\mathbf{v}) = v_k - v_l$, putting together the above results concludes. □

We are now able to answer the first question about the existence of a matrix $\mathbf{U} \in \mathbb{U}^d$ for two specific values of T.

Lemma 2. *Let* T *be either equal to* $\{\mathbf{1}\}$ *or* $\{\mathbf{e}_1, \dots, \mathbf{e}_d\}$*. If* $\mathbf{v} \in \Omega(T) \setminus T$*, then there exists* $\mathbf{U} \in \mathbb{U}^d$ *such that* $\mathbf{U}\mathbf{v} \neq \mathbf{v}$ *and* $\mathbf{U}\mathbf{v} \in \Omega(T)$*.*

Proof. Since there are different corner cases for the two possible values of T, we separately deal with them.

Case $T = \{\mathbf{1}\}$*:* If $\mathbf{v} \in \Omega(T) \setminus T$, there are two consequences. First, $\mathbf{1} \leq \mathbf{v}$, which means that the coordinates of $\mathbf{v}$ are greater than or equal to 1. Second, $\mathbf{v} \in \mathbb{P}^d$ and $\mathbf{v} \neq \mathbf{1}$, which means that the coordinates of $\mathbf{v}$ are not all equal. As a result, there exist two disinct $k, l \in \{1, \dots, d\}$ such that $v_l \geq 1$ and $v_k - v_l \geq 1$.

Case $T = \{\mathbf{e}_1, \dots, \mathbf{e}_d\}$*:* Again, if $\mathbf{v} \in \Omega(T) \setminus T$, there are two consequences, but slightly different. First, there is $\mathbf{t} \in \{\mathbf{e}_1, \dots, \mathbf{e}_d\}$ such that $\mathbf{t} \leq \mathbf{v}$, which means that all coordinates of $\mathbf{v}$ are greater than or equal to 0. Second, $\mathbf{v} \in \mathbb{P}^d$ and $\mathbf{v} \neq \mathbf{t}$, which means that $\mathbf{v}$ contains at least two nonzero coordinates. As a result, there exist two disinct $k, l \in \{1, \dots, d\}$ such that $v_l \geq 1$ and $v_k - v_l \geq 0$.

In both cases, using Lemma 1, we conclude that there exists a matrix $\mathbf{U} \in \mathbb{U}^d$, whose entry (k, l) equals -1, such that $\mathbf{U}\mathbf{v} \neq \mathbf{v}$ and $\mathbf{U}\mathbf{v} \in \Omega(T)$. □

We are now able to study the termination and complexity of Algorithm 1 for two specific values of T.

Theorem 3. *Let* T *be either equal to* $\{\mathbf{1}\}$ *or* $\{\mathbf{e}_1, \dots, \mathbf{e}_d\}$*. Let* ω *be equal to the* 1*-norm of the input vector* $\mathbf{v}$*. Algorithm 1 terminates after less than* ω *iterations.*

Proof. The coordinates of $\mathbf{v}$ are always nonnegative and the matrix multiplication consists in subtracting a nonzero coordinate from another. The integral number $\|\mathbf{v}\|_1$ thus strictly decreases throughout the iterations. In addition, $\mathbf{v}$ is always in $\Omega(T)$ by Lemma 2. The algorithm stops when $\mathbf{v}$ reaches a bottom element in $\Omega(T)$, i.e., an element of T. Consequently, the algorithm terminates when $\|\mathbf{v}\|_1 = d$ if $T = \{\mathbf{1}\}$ or when $\|\mathbf{v}\|_1 = 1$ if $T = \{\mathbf{e}_1, \dots, \mathbf{e}_d\}$, after less than ω iterations. □

In the rest of the paper, we will mainly focus on the following finite sequences:

Definition 1. *A sequence of matrices* $(\mathbf{U}_n)_{0 \leq n \leq N}$ *is* valid *if and only if* $\mathbf{U}_0 = \mathbf{I}_d$ *and every* $\mathbf{U}_n$*,* $n \in \{1, \dots, N\}$*, is in* $\mathbb{U}^d$*. In addition, a valid sequence* reduces *a vector* $\mathbf{v}$ *if and only if there exists* $T \subset \mathbb{P}^d$ *such that* $\mathbf{U}_n \cdots \mathbf{U}_0\mathbf{v} \in \Omega(T) \setminus T$ *for all* $n \in \{0, \dots, N-1\}$ *and* $\mathbf{U}_N \cdots \mathbf{U}_0\mathbf{v} \in T$*. In that case, we set* $\mathbf{v}_n := \mathbf{U}_n \cdots \mathbf{U}_0\mathbf{v}$ *and* $\mathbf{a}_n := \mathbf{U}_0^{-1} \cdots \mathbf{U}_n^{-1}\mathbf{v}_N$*.*

By design and by Theorem 3, Algorithm 1 returns a sequence of matrices that reduces the input vector $\mathbf{v}$. Furthermore, the output of the algorithm provides a way of representing the input thanks to the following equalities:

$$\mathbf{a}_N = \mathbf{U}_0^{-1} \cdots \mathbf{U}_N^{-1}\mathbf{v}_N = \mathbf{U}_0^{-1} \cdots \mathbf{U}_N^{-1}(\mathbf{U}_N \cdots \mathbf{U}_0\mathbf{v}) = \mathbf{v}. \tag{1}$$

Note that $\mathbf{a}_n$ is generally not equal to $\mathbf{v}_n$ and, for $n \neq N$, is not equal to $\mathbf{v}$.

2.3 Examples in low dimensions

In this subsection, we show how Algorithm 1 relates with the Euclidean algorithm in 2D and some of its well-known generalizations in 3D.

2.3.1 In 2D.

The set $\mathbb{U}^2$ only contains two matrices:

$$\mathbf{L} := \left(\begin{smallmatrix} 1 & -1 \\ 0 & 1 \end{smallmatrix}\right) \text{ and } \mathbf{R} := \left(\begin{smallmatrix} 1 & 0 \\ -1 & 1 \end{smallmatrix}\right).$$

When fed up with the terminal set $T = \{\mathbf{1}\}$ and the input vector $\mathbf{v}$, Algorithm 1 does the following:

- if $v_1 > v_2$, then $\mathbf{v} \leftarrow \mathbf{Lv}$ and repeat,
- if $v_1 < v_2$, then $\mathbf{v} \leftarrow \mathbf{Rv}$ and repeat,
- otherwise, $\mathbf{v}$ must be equal to $\mathbf{1}$, stop and return,

which is exactly the Euclidean algorithm in its additive form.

For instance, with $T = \{\mathbf{1}\}$ and ${}^{\mathrm{t}}\mathbf{v} = (5, 3)$ as input, Algorithm 1 returns the list $(\mathbf{I}_2, \mathbf{L}, \mathbf{R}, \mathbf{L})$. Hence the following equivalence:

$$\mathbf{1} = \mathbf{LRL}\left(\begin{smallmatrix} 5 \\ 3 \end{smallmatrix}\right) \Leftrightarrow \mathbf{L}^{-1}\mathbf{R}^{-1}\mathbf{L}^{-1}\mathbf{1} = \left(\begin{smallmatrix} 5 \\ 3 \end{smallmatrix}\right).$$

Note that using the terminal set $T = \{\mathbf{e}_1, \mathbf{e}_2\}$ instead of $T = \{\mathbf{1}\}$ results in replacing $\mathbf{1}$ with $\mathbf{L}^{-1}\mathbf{e}_2$.

2.3.2 In 3D.

In dimension 2, it is clear to decide which coordinate has to be subtracted to the other, whereas in dimension 3, it is not the case, hence the diversity of existing generalizations. See [17] for a list as well as a detailed analysis of several of them. Several existing algorithms stick to a convention. For instance, assuming w.l.o.g. $0 \le v_1 \le v_2 \le v_3$, we have:

- *Brun*:
$$\mathbf{v} \mapsto \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix}\mathbf{v},$$
- *Selmer*:
$$\mathbf{v} \mapsto \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}\mathbf{v}.$$

In both strategies, the subtraction is encoded in a matrix $\mathbf{U} \in \mathbb{U}^3$. Furthermore, for any vector of coprime and strictly positive coordinates, Selmer can be reproduced using Algorithm 1 and $T = \{\mathbf{1}\}$. In order to be able to deal with null coordinates, it is enough to choose $T = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$. Brun can also be reproduced using Algorithm 1, but only with $T = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ because a zero coordinate may appear whenever the two largest coordinates are equal.

For instance, with $T = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ and ${}^{\mathrm{t}}\mathbf{v} = (1, 2, 2)$ as input, using Brun in line 3 for the matrix selection and taking the index order in case of ties, Algorithm 1 returns the terminal $\mathbf{e}_1$ and the following list:

$$\mathbf{U}_0 = \mathbf{I}_3, \quad \mathbf{U}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{U}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}, \quad \mathbf{U}_3 = \mathbf{U}_2.$$

Hence the following equivalence:

$$\mathbf{e}_1 = \mathbf{U}_3\mathbf{U}_2\mathbf{U}_1\mathbf{U}_0\begin{pmatrix} 1 \\ 2 \\ 2 \end{pmatrix} \Leftrightarrow \mathbf{U}_0^{-1}\mathbf{U}_1^{-1}\mathbf{U}_2^{-1}\mathbf{U}_3^{-1}\mathbf{e}_1 = \begin{pmatrix} 1 \\ 2 \\ 2 \end{pmatrix}.$$

Other less elementary strategies may be used as well. For instance,

- *Poincaré*: let $0 \leq v_1 \leq v_2 \leq v_3$,
$$\mathbf{v} \mapsto \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix} \mathbf{v},$$
- *Jacobi-Perron*: let $0 < v_1, v_2 \leq v_3$ and $q_2 = [\frac{v_2}{v_1}], q_3 = [\frac{v_3}{v_1}]$,
$$\mathbf{v} \mapsto \begin{pmatrix} 1 & 0 & 0 \\ -q_2 & 1 & 0 \\ -q_3 & 0 & 1 \end{pmatrix} \mathbf{v}.$$

One can check that we can choose $T = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ as terminal set for these two strategies by possibly applying the two-dimensional Euclidean algorithm once a coordinate has reached the value 0 in the Jacobi-Perron case. Even if the above matrices are not in $\mathbb{U}^3$, one can check that they can be replaced by a product of elementary matrices, which are all in $\mathbb{U}^3$. Algorithm 1 is thus also a generalization of such strategies.

2.4 Relation with Digital Geometry

The goal of this subsection is to relate Algorithm 1 to algorithms recently developed in the context of digital geometry [20, 21]. This relation is based on the transpose of the partial product $\mathbf{U}_n \dots \mathbf{U}_0$, i.e., ${}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n$. Even if the above-mentioned algorithms are three-dimensional, the following propositions are valid in a d-dimensional space. They gather several properties involving the matrices ${}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n$ as well as $\mathbf{v}_n$ and $\mathbf{a}_n$ introduced in Definition 1:

Proposition 4. *For all $n \in \{0, \dots, N\}$ and $i \in \{1, \dots, d\}$:*

(i) ${}^{t}\mathbf{v}_0\, {}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n \mathbf{e}_i = {}^{t}\mathbf{e}_i \mathbf{v}_n$,

(ii) ${}^{t}\mathbf{a}_n\, {}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n \mathbf{e}_i = {}^{t}\mathbf{e}_i \mathbf{v}_N$,

Proof. In both cases, it is enough to use the definitions of $\mathbf{v}_n$ and $\mathbf{a}_n$:

(i) $\left({}^{t}\mathbf{v}_0\, {}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n\right)\mathbf{e}_i = {}^{t}\mathbf{e}_i\left(\mathbf{U}_n \cdots \mathbf{U}_0 \mathbf{v}_0\right) = {}^{t}\mathbf{e}_i \mathbf{v}_n$.

(ii) $\left({}^{t}\mathbf{a}_n\, {}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n\right)\mathbf{e}_i = {}^{t}\mathbf{e}_i\left(\mathbf{U}_n \cdots \mathbf{U}_0(\mathbf{U}_0^{-1} \cdots \mathbf{U}_n^{-1}\mathbf{v}_N)\right) = {}^{t}\mathbf{e}_i \mathbf{v}_N$.

□

Definition 2. *For all $n \in \{0, \dots, N\}$, let $\Lambda_n := \{\mathbf{x} \in \mathbb{Z}^d \mid {}^{t}\mathbf{a}_n \mathbf{x} = 0\}$ and let B_n be defined as follows:*

- $B_n := \left({}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n(\mathbf{e}_i - \mathbf{e}_{i-1})\right)_{i \in \{2, \dots, d\}}$ *if* $\mathbf{v}_N = \mathbf{1}$,
- $B_n := \left({}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n \mathbf{e}_i\right)_{i \in \{2, \dots, d\}}$ *if* $\mathbf{v}_N \in \{\mathbf{e}_1, \dots, \mathbf{e}_d\}$ *and assuming w.l.o.g. that* $\mathbf{v}_N = \mathbf{e}_1$.

Proposition 5. *The $(d-1)$ tuple B_n is a basis of the lattice Λ_n.*

Proof. (Sketch) For both definitions of B_n, it is enough to notice that the dot product between $\mathbf{a}_n$ and each vector of the tuple is equal to 0 by Proposition 4, item (ii). The fact that the tuple is a *basis* of the lattice Λ_n comes from the fact that the matrix ${}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n$ has determinant 1. □

Fig. 2 illustrates the action of the matrix ${}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_N$. Note that for each $n \in \{0, \dots, N\}$, the matrix ${}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n$ transforms the basis $\{\mathbf{e}_i \mid i \in \{1, 2, 3\}\}$ into $\{{}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n \mathbf{e}_i \mid i \in \{1, 2, 3\}\}$. In other words, it deforms an orthonormal basis to a basis that is more and more aligned with the plane of normal $\mathbf{v}_0$ because the sum of the quantities $\{{}^{t}\mathbf{v}_0\, {}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_n \mathbf{e}_i \mid i \in \{1, 2, 3\}\}$ is smaller and smaller by Proposition 4, item (i). At the last step, the i-th column of ${}^{t}\mathbf{U}_0 \cdots {}^{t}\mathbf{U}_N$ is perfectly aligned with the plane of normal $\mathbf{v}_0$ when ${}^{t}\mathbf{e}_i \cdot \mathbf{v}_N = 0$, but not when ${}^{t}\mathbf{e}_i \cdot \mathbf{v}_N = 1$. Despite this, one can always deduce $d-1$ vectors perfectly aligned with the plane of normal $\mathbf{v}_0$ by Proposition 5.

The next proposition requires the following definition.

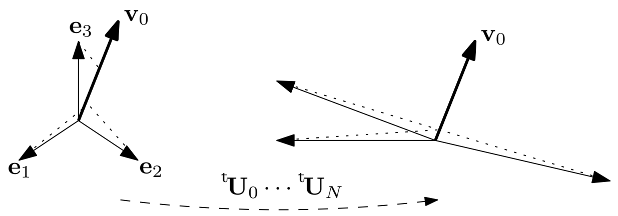


Figure 2: Geometrical interpretation of the matrix ${}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_N$. We have implicitly represented the scalar projection of the basis vectors in the direction of $\mathbf{v}_0$ with the help of dotted lines. The scalar products $\{{}^{\mathrm{t}}\mathbf{v}_0\, {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_N \mathbf{e}_i \mid i \in \{1,2,3\}\}$ are equal to the coordinates of $\mathbf{v}_N$ and are thus all equal to 0 or 1.

Definition 3. *The* digital hyperplane *of normal* $\mathbf{v} \in \mathbb{P}^d$ *is the infinite set:*

$$\mathcal{P}_{\mathbf{v}} := \{\mathbf{x} \in \mathbb{Z}^d \mid 0 \leq {}^{\mathrm{t}}\mathbf{v}\mathbf{x} < \|\mathbf{v}\|_1\}. \tag{2}$$

Proposition 6. *For all* $n \in \{0, \dots, N\}$ *and* $i \in \{1, \dots, d\}$*:*

(i) if $\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \in \mathcal{P}_{\mathbf{v}_0}$*, then* ${}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n \geq 1$.

(ii) if $\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \notin \mathcal{P}_{\mathbf{v}_0}$ *and* $\mathbf{1} + {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \notin \mathcal{P}_{\mathbf{v}_0}$*, then* ${}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n = 0$.

Proof. We first have

$${}^{\mathrm{t}}\mathbf{v}_0(\mathbf{1} \pm {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i) = {}^{\mathrm{t}}\mathbf{v}_0 \mathbf{1} \pm {}^{\mathrm{t}}\mathbf{v}_0({}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i) = \|\mathbf{v}_0\|_1 \pm {}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n,$$

where the last equality comes from ${}^{\mathrm{t}}\mathbf{v}_0 \mathbf{1} = \|\mathbf{v}_0\|_1$ and Proposition 4, item (i).

By Definition 3, $\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \in \mathcal{P}_{\mathbf{v}_0}$ implies

$${}^{\mathrm{t}}\mathbf{v}_0(\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i) < \|\mathbf{v}_0\|_1 \Leftrightarrow \|\mathbf{v}_0\|_1 - {}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n < \|\mathbf{v}_0\|_1 \Leftrightarrow {}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n > 0,$$

which proves (i).

We now focus on (ii). By Definition 3, $\mathbf{1} \pm {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \notin \mathcal{P}_{\mathbf{v}_0}$ implies either

$${}^{\mathrm{t}}\mathbf{v}_0(\mathbf{1} \pm {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i) \geq \|\mathbf{v}_0\|_1 \text{ or } {}^{\mathrm{t}}\mathbf{v}_0(\mathbf{1} \pm {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i) < 0.$$

The second inequality is equivalent to $\pm\, {}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n > \|\mathbf{v}_0\|_1$, which is false because $\pm\, {}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n \leq \|\mathbf{v}_n\|_1 \leq \|\mathbf{v}_0\|_1$. As a result, only the first inequality, which is equivalent to $\pm\, {}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n \leq 0$, is true. We thus clearly get ${}^{\mathrm{t}}\mathbf{e}_i \mathbf{v}_n = 0$ if we have both $\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \notin \mathcal{P}_{\mathbf{v}_0}$ and $\mathbf{1} + {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \notin \mathcal{P}_{\mathbf{v}_0}$. □

Let us now consider the predicate InPlane := "is $\mathbf{x}$ in $\mathcal{P}$?" for any hyperplane $\mathcal{P} \in \{\mathcal{P}_{\mathbf{v}} \mid \mathbf{v} \in \mathbb{P}^d\}$ of unknown normal vector. Proposition 6 shows that we are able to check if a given coordinate of the unknown normal vector is strictly positive or equal to 0 by probing the hyperplane with the predicate InPlane. By Lemma 1, such tests on coordinates are a way of selecting a possible matrix on line 3 of Algorithm 1. As a consequence, a geometrical version of Algorithm 1 can be obtained by replacing the vector $\mathbf{v}$ by the predicate InPlane and changing the way a new matrix is selected accordingly. The three-dimensional *plane-probing* algorithms H, R [20] and L [21] rely on exactly that idea.

The difference between the three algorithms lies in their set of possible operations. Those operations can be represented up to symmetries by a matrix:

$$\mathbf{v} \mapsto \begin{pmatrix} 1 & -a & -b \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{v}.$$

In H, a is set to 1 and b to 0, which means that the matrix belongs to $\mathbb{U}^3$. In R, a is also set to 1, but b can be any nonnegative integer. In L, both a and b can be any nonnegative integers, provided that they are not both null. The matrices used in R and L are not necessarily in $\mathbb{U}^3$. However, if they are not, they can be replaced by a product of elementary matrices, which are all in $\mathbb{U}^3$. We can therefore assume that H, R and L produce a valid sequence of matrices $(\mathbf{U})_{0 \leq n \leq N}$. In addition, that sequence reduces the normal vector of the underlying hyperplane because:

- $\forall n \in \{0, \dots, N-1\}$, $\mathbf{U}_n \cdots \mathbf{U}_0 \mathbf{v} \in \Omega(\{\mathbf{1}\}) \setminus \{\mathbf{1}\}$ by item 1 of [20, Lemma 1],
- $\mathbf{U}_N \cdots \mathbf{U}_0 \mathbf{v} = \mathbf{1}$ by [20, Theorem 2].

As in (1), we have $\mathbf{a}_N = \mathbf{v}$ [20, Corollary 4]. In other words, those algorithms, which compute $\mathbf{a}_N$ only from the predicate InPlane, are indeed able to retrieve the normal vector $\mathbf{v}$ of $\mathcal{P}$.

To end the section, note that one can even further generalize our approach if one replaces InPlane by another set-membership predicate involving a given digital surface as shown in Fig. 1 and [20, Section 5].

2.5 Discussion

In this subsection, we compare the plane-probing algorithms H, R and L to classical three-dimensional Euclidean algorithms, in particular, Brun, Selmer, Poincaré and Jacobi-Perron presented above.

What makes those two kinds of algorithms very different from each other lies in the way they select a matrix in line 3 of Algorithm 1. On one hand, plane-probing algorithms use a *geometrical criterion* involving the column vectors of ${}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n$. On the other hand, classical three-dimensional Euclidean algorithms, even when run with the predicate InPlane, use an *arithmetical criterion* involving the coordinates of $\mathbf{U}_n \cdots \mathbf{U}_0 \mathbf{v}$.

Non-uniqueness. In general, several matrices can be chosen. The arithmetical criterion aims at reducing the set to one, but if $\mathbf{v}_n$ has equal coordinates, there is still more than one possible matrix to get $\mathbf{v}_{n+1}$. An often-used convention to break ties is to compare the indices of the equal coordinates. The geometrical criterion is based on an in-sphere test [20] and usually reduces the set to one, but in case of co-spherical points, again, there is more than one possible matrix. In such cases, one can resort to a lexicographic order.

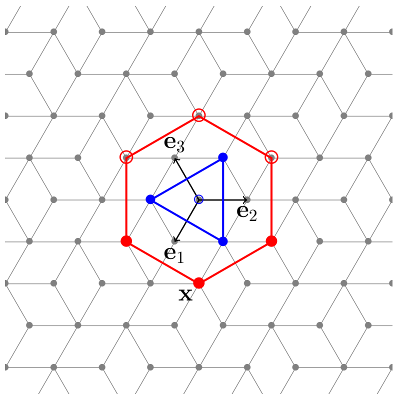
Fig. 3 illustrates the first steps when applying Selmer (top row) and H (bottom row) on ${}^{\mathrm{t}}\mathbf{v} = (2, 3, 5)$. If the point $\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n(\mathbf{e}_k - \mathbf{e}_l)$, with distinct $k, l \in \{1, 2, 3\}$, belongs to $\mathcal{P}$ – such a point is depicted with a red disk in Fig. 3 –, then there exists a unique matrix $\mathbf{U} \in \mathbb{U}^3$ containing -1 at the intersection of the k-th row and the l-th column such that

$$\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n(\mathbf{e}_k - \mathbf{e}_l) = \mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n\, {}^{\mathrm{t}}\mathbf{U}\mathbf{e}_k \in \mathcal{P}.$$

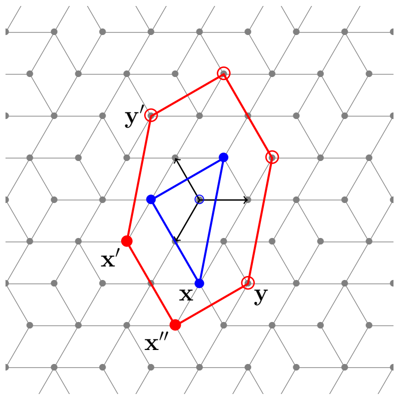
As a result, by Proposition 6 and Lemma 1, $\mathbf{U}$ is a possible candidate for $\mathbf{U}_{n+1}$. With that geometrical point of view, in Fig. 3, choosing a red disk is like choosing a matrix.

Selmer first selects $\mathbf{x}$ (see (a) and (b) in Fig. 3). Then, an equality between two coordinates of $\mathbf{v}_1$ is detected, because there are two points symmetric about $\mathbf{1}$, namely $\mathbf{y}$ and $\mathbf{y}'$, that do not belong to $\mathcal{P}$. From the two candidate points $\mathbf{x}' = \mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_2$ and $\mathbf{x}'' = \mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_3$, the second one is chosen because it has the greatest index: 3 (see (b) and (c) in Fig. 3).

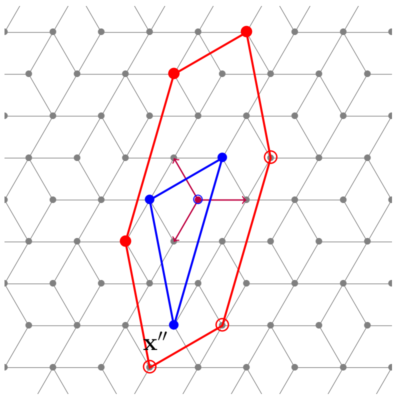
Regarding H, the red and blue disks are in a co-spherical position at the very beginning, but $\mathbf{z}$ is chosen according to the lexicographic order on points (see (d) and (e) in Fig. 3). Then, $\mathbf{z}''$ is chosen because the sphere passing by the triangle and $\mathbf{z}''$ does not contain $\mathbf{z}'$ (see (e) and (f) in Fig. 3).



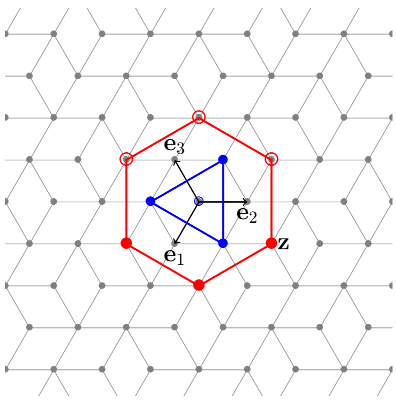
(a) ${}^{\mathrm{t}}\mathbf{v}_0 = (2, 3, 5)$



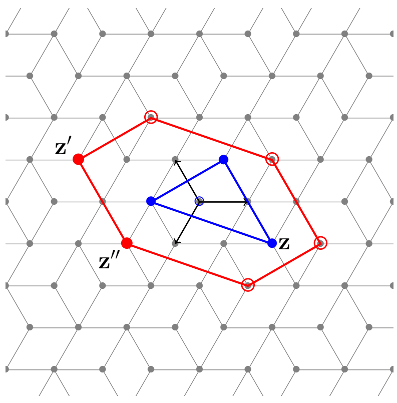
(b) ${}^{\mathrm{t}}\mathbf{v}_1 = (2, 3, 3)$



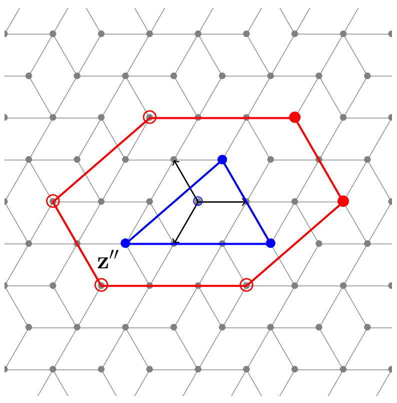
(c) ${}^{\mathrm{t}}\mathbf{v}_2 = (2, 3, 1)$



(d) ${}^{\mathrm{t}}\mathbf{v}_0 = (2, 3, 5)$



(e) ${}^{\mathrm{t}}\mathbf{v}_1 = (2, 3, 2)$



(f) ${}^{\mathrm{t}}\mathbf{v}_2 = (2, 1, 2)$

Figure 3: The digital plane of normal ${}^{\mathrm{t}}\mathbf{v} = (2, 3, 5)$ is depicted in gray. The vector $\mathbf{v}$ has been reduced with Selmer in (a)-(c) and H in (d)-(f). Only the steps $n = 0, 1, 2$ are shown. For every step, the three-point tuple $(\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i)_{i \in \{1,2,3\}}$ is depicted as a blue triangle, while the set of differences $(\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n(\mathbf{e}_k - \mathbf{e}_l))_{k,l \in \{1,2,3\}, k \neq l}$ is depicted as a red hexagon, which is a usual representation for plane-probing algorithms. The vertices are depicted with disks (resp. circles) if they belong (resp. do not belong) to the digital plane.

Impact of previous choices. In the arithmetical approach, the choice of $\mathbf{U}_{n+1}$ only depends on the coordinates of $\mathbf{v}_n$, regardless of how $\mathbf{v}_n$ has been obtained. If, reducing two different input vectors $\mathbf{v}$ and $\mathbf{v}'$, one has $\mathbf{v}_n = \mathbf{v}'_{n'}$, the next matrices will be exactly the same for the two sequences. This is not the case in the geometrical approach, because the choice of $\mathbf{U}_{n+1}$ depends on the whole sequence. In that approach, even if $\mathbf{v}_n$ equals $\mathbf{v}'_{n'}$, the next matrices will be in general different for the two reductions.

Geometrical properties. The column vectors of ${}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n$ exhibit interesting properties in the geometrical approach. One such property is that the basis of the rank-two lattice Λ_n returned by R (resp. H) is experimentally always (resp. almost always) reduced [20]. It has even been proven that the basis returned by L is always reduced [24]. Another property concerns the points selected during the execution of the algorithm, i.e., the set:

$$\{\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \dots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i \mid n \in \{0, \dots, N\}, i \in \{1, \dots, d\}\}. \tag{3}$$

Experimentally, those points are in average much closer to $\mathbf{1}$ in the geometrical approach than in the arithmetical approach. This observation is illustrated in Fig. 4 and supported by the experiments described below.

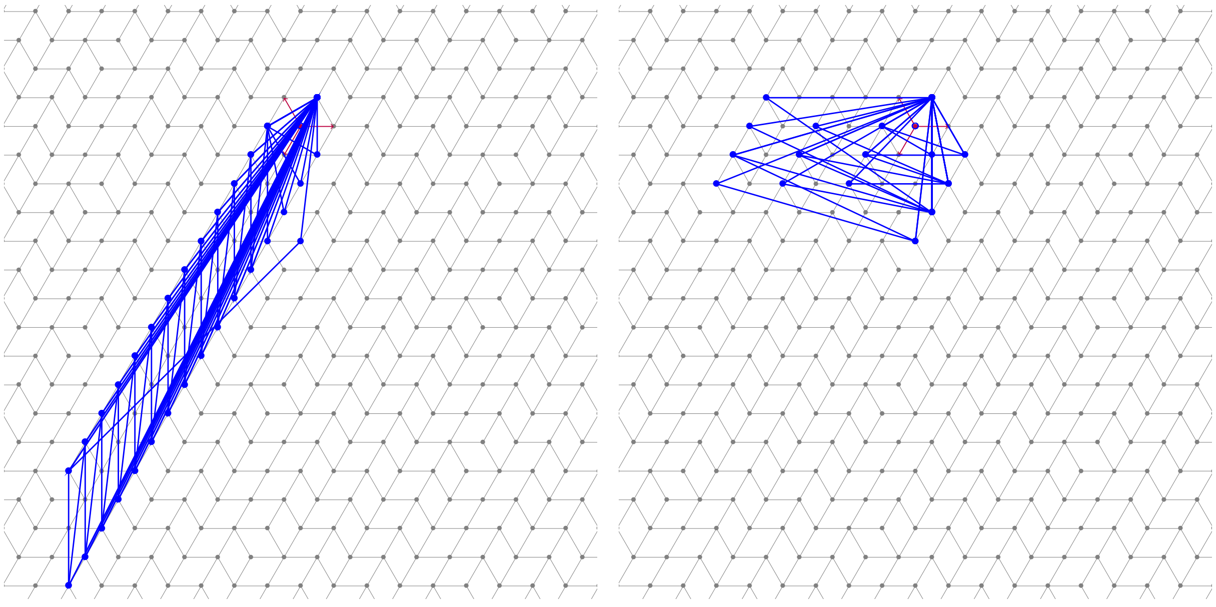

(a) Selmer (b) H

Figure 4: The digital plane of normal ${}^{\mathrm{t}}\mathbf{v} = (1, 13, 17)$ is depicted in gray. The vector $\mathbf{v}$ has been reduced with Selmer in (a) and H in (b). For every step n, the three-point tuple $(\mathbf{1} - {}^{\mathrm{t}}\mathbf{U}_0 \cdots {}^{\mathrm{t}}\mathbf{U}_n \mathbf{e}_i)_{i \in \{1,2,3\}}$ is depicted as a blue triangle.

Experimental results. As in [20, Table 1], we ran all algorithms on vectors ranging from $(1, 1, 1)$ to $(200, 200, 200)$. There are 6578833 vectors with relatively prime components in that range. For all of them, we checked whether the returned basis is reduced or not. In the latter case, we also computed the number of reductions needed to reduce it *a posteriori*. Results are reported in Table 1. The returned bases are generally not reduced with the arithmetical approach, while they generally are with the geometrical approach.

Furthermore, we computed for all vectors the maximal distance between $\mathbf{1}$ and every point of the set described in (3). In Fig. 5, we plotted that maximal distance against the magnitude of the input vector. In order to smooth the lines, we discretized the magnitude of the input vector into bins of size 10 and took the average for every bin. The lines for H, R and L lie upon each

Algorithm	H	R	L	B	S	P	JP
non-reduced (%)	0.01	0	0	93.37	84.16	96.18	97.29
avg. reductions	1.00			5.27	5.43	481.68	7.73

Table 1: All algorithms were run on all vectors ranging from $(1, 1, 1)$ to $(200, 200, 200)$ with relatively prime coordinates. Besides H, R and L, other algorithms are denoted by their initials. The average number of reductions is computed only among non-reduced bases.

other because those three algorithms often make the same choices for the considered vectors. More importantly, those lines lie quite below the other ones, which means that the geometrical approach is much more local than the arithmetical one, as suggested by Fig. 4.

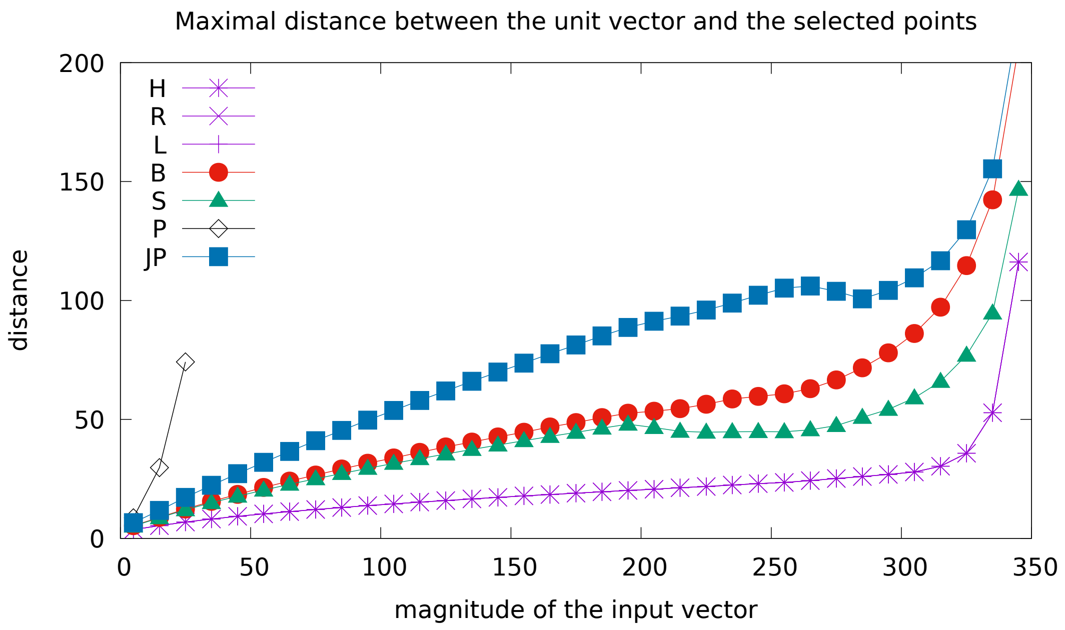


Figure 5: The maximal distance between **1** and every point selected during the execution of the algorithm is plotted against the magnitude of the input vector (see the text for the experimental settings). The line labeled by P is truncated, because of its extremely large slope.

3 Pattern Generation with Extensions of Substitutions

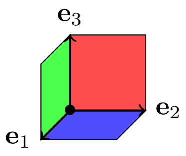
As discussed in the previous section, Algorithm 1 provides a terminal and a sequence of matrices that reduces the input vector $\mathbf{v}$. A variant consists in passing in the predicate InPlane instead of $\mathbf{v}$ as with plane-probing algorithms. In this section, we show how to generate a piece of digital hyperplane of normal $\mathbf{v}$ from a sequence of matrices returned by Algorithm 1 or its variant. Our method is based on a description of digital hyperplanes as unions of faces, recalled in Subsection 3.1. The mathematical framework we use is presented in Subsection 3.2. It provides us with a clear definition from which one can derive a generation method in Subsection 3.3. The properties of the generated sets are then discussed in the last two subsections.

3.1 Digital Hyperplanes as Unions of Faces

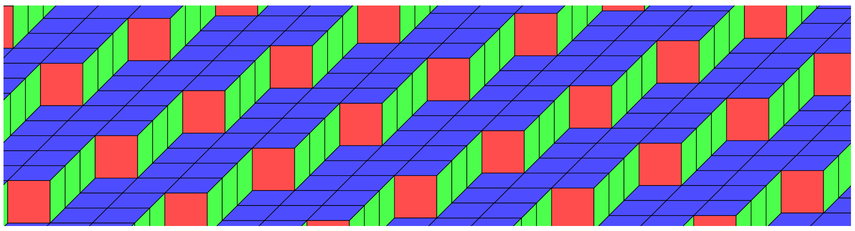
For $\mathbf{x} \in \mathbb{Z}^d$ and $i \in \{1, \dots, d\}$, we define the $(d-1)$-dimensional face in $\mathbf{x}$ perpendicular to the direction i as the following subset of $\mathbb{R}^3$:

$$(\mathbf{x}, i^*) := \left\{ \mathbf{x} + \sum_{j \in \{1,\dots,d\} \setminus i} \lambda_j \mathbf{e}_j \mid \lambda_j \in [0,1] \right\}. \tag{4}$$

The point $\mathbf{x}$ will be called the *origin* of the face and i the *type* of the face. A three-dimensional illustration is given in Fig. 6 (a).



(a)



(b)

Figure 6: In (a), illustration of two-dimensional faces embedded in a three-dimensional space. The faces $(\mathbf{0}, 1^*), (\mathbf{0}, 2^*), (\mathbf{0}, 3^*)$ are respectively in red, green and blue. In (b), the stepped plane of normal ${}^{\mathrm{t}}(2, 6, 15)$ is displayed.

Definition 4 (Fig. 6 (b)). *The* stepped hyperplane *of normal* $\mathbf{v} \in \mathbb{P}^d$ *is the infinite set:*

$$\mathcal{S}_{\mathbf{v}} := \{(\mathbf{x}, i^*) \in \mathbb{Z}^d \times \{1, \dots, d\} \mid 0 \le {}^{\mathrm{t}}\mathbf{v}\mathbf{x} < {}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i\}. \tag{5}$$

The link between digital hyperplanes (Definition 3) and stepped hyperplanes (Definition 4) is made explicit in the following proposition.

Proposition 7. *Every face in $\mathcal{S}_{\mathbf{v}}$ have its vertices in $\mathcal{P}_{\mathbf{v}}$. Conversely, every point in $\mathcal{P}_{\mathbf{v}}$ is a vertex of a face in $\mathcal{S}_{\mathbf{v}}$.*

Proof. Let us introduce the following set of partial sums:

$$\forall i \in \{1, \dots, d\}, \mathbb{T}_i := \left\{ \sum_{j \in J} \mathbf{e}_j \mid J \subseteq \{1, \dots, d\} \setminus i \right\}.$$

Note that the vertices of a face $(\mathbf{x}, i^*) \in \mathcal{S}_{\mathbf{v}}$ form the set $\{\mathbf{x} + \mathbf{t} \mid \mathbf{t} \in \mathbb{T}_i\}$.

We first show that all the vertices of a face $(\mathbf{x}, i^*) \in \mathcal{S}_{\mathbf{v}}$ are in $\mathcal{P}_{\mathbf{v}}$. By definition $(\mathbf{x}, i^*) \in \mathcal{S}_{\mathbf{v}}$ is equivalent to $0 \le {}^{\mathrm{t}}\mathbf{v}\mathbf{x} < {}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i$. By adding ${}^{\mathrm{t}}\mathbf{v}\mathbf{t}, \mathbf{t} \in \mathbb{T}_i$, we get ${}^{\mathrm{t}}\mathbf{v}\mathbf{t} \le {}^{\mathrm{t}}\mathbf{v}(\mathbf{x} + \mathbf{t}) < {}^{\mathrm{t}}\mathbf{v}(\mathbf{e}_i + \mathbf{t})$. In addition, we have by definition $0 \le {}^{\mathrm{t}}\mathbf{v}\mathbf{t}$ and ${}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i + {}^{\mathrm{t}}\mathbf{v}\mathbf{t} < {}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i + {}^{\mathrm{t}}\mathbf{v}\big(\sum_{j \in \{1,\dots,d\} \setminus i} \mathbf{e}_j\big) = {}^{\mathrm{t}}\mathbf{v}\mathbf{1} = \|\mathbf{v}\|_1$. We have therefore $0 \le {}^{\mathrm{t}}\mathbf{v}(\mathbf{x} + \mathbf{t}) < \|\mathbf{v}\|_1$, i.e., $\mathbf{x} + \mathbf{t} \in \mathcal{P}_{\mathbf{v}}$ for all $\mathbf{t} \in \mathbb{T}_i$ by (2).

Now, we show that for all points $\mathbf{y} \in \mathcal{P}_{\mathbf{v}}$, there is a face of type i in $\mathcal{S}_{\mathbf{v}}$ such that $\mathbf{y}$ is one of its vertices. More precisely, we choose i such that ${}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i$ is maximal, i.e., $i = \arg\max_{j \in \{1,\dots,d\}} {}^{\mathrm{t}}\mathbf{v}\mathbf{e}_j$. In addition, we number the K partial sums of $\mathbb{T}_i$ according to the following order: ${}^{\mathrm{t}}\mathbf{v}\mathbf{t}_1 \le \dots \le {}^{\mathrm{t}}\mathbf{v}\mathbf{t}_k \le \dots \le {}^{\mathrm{t}}\mathbf{v}\mathbf{t}_K$.

Note that $\mathbf{t}_K = \sum_{j \in \{1,\dots,d\} \setminus i} \mathbf{e}_j$ and, because of our choice for i, ${}^{\mathrm{t}}\mathbf{v}\mathbf{t}_{k+1} - {}^{\mathrm{t}}\mathbf{v}\mathbf{t}_k \le {}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i$ for all $k \in \{1, \dots, K-1\}$.

If ${}^{\mathrm{t}}\mathbf{v}\mathbf{t}_K \le {}^{\mathrm{t}}\mathbf{v}\mathbf{y} < \|\mathbf{v}\|_1$, then $0 \le {}^{\mathrm{t}}\mathbf{v}(\mathbf{y} - \mathbf{t}_K) < \|\mathbf{v}\|_1 - {}^{\mathrm{t}}\mathbf{v}\mathbf{t}_K = {}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i$, i.e. $(\mathbf{y} - \mathbf{t}_K, i^*) \in \mathcal{S}_{\mathbf{v}}$ by (5).

For all $k \in \{1, \dots, K-1\}$, if ${}^{\mathrm{t}}\mathbf{v}\mathbf{t}_k \le {}^{\mathrm{t}}\mathbf{v}\mathbf{y} < {}^{\mathrm{t}}\mathbf{v}\mathbf{t}_{k+1}$, then $0 \le {}^{\mathrm{t}}\mathbf{v}(\mathbf{y} - \mathbf{t}_k) < {}^{\mathrm{t}}\mathbf{v}\mathbf{t}_{k+1} - {}^{\mathrm{t}}\mathbf{v}\mathbf{t}_k \le {}^{\mathrm{t}}\mathbf{v}\mathbf{e}_i$, i.e. $(\mathbf{y} - \mathbf{t}_k, i^*) \in \mathcal{S}_{\mathbf{v}}$ by (5). $\square$

3.2 Words, Substitutions and their Extensions

In this subsection, we present the framework we use for our generation method. It is based on geometrical extensions of substitutions: rules that replace faces by unions of faces in the same way that substitutions replace letters by words.

We consider a d-letter alphabet $\mathcal{A}_d := \{1, \ldots, d\}$. A *word* is an element of the free monoid $\mathcal{A}_d^\star$ generated by $\mathcal{A}_d$. The empty word is denoted by ϵ and the concatenation operation is denoted by $\cdot$ or is left implicit. The *abelianization mapping* $f : \mathcal{A}_d^\star \to \mathbb{N}^d$ is such that ${}^{\mathrm{t}}f(w) = (|w|_1, |w|_2, |w|_3)$, where $|w|_i$ denotes the number of occurrences of the letter i in w.

Example 1. *The word* $w = 1 \cdot 2 \cdot 3 \cdot 1 \cdot 2 = 12312$ *is an element of* $\{1, 2, 3\}^\star$. ${}^{\mathrm{t}}f(w) = (2, 2, 1)$.

A *substitution* σ over $\mathcal{A}_d$ is a non-erasing endomorphism of $\mathcal{A}_d^\star$, completely defined by its image on the letters of $\mathcal{A}_d$ by the relation $\sigma(w \cdot w') = \sigma(w) \cdot \sigma(w')$. The *incidence matrix* $\mathbf{M}_\sigma$ of σ is the $d \times d$ matrix whose i-th column is $f(\sigma(i))$ for every $i \in \mathcal{A}_d$. For any word $w \in \mathcal{A}_d^\star$, applying σ then f is the same as applying f then $\mathbf{M}_\sigma$, i.e., $\mathbf{M}_\sigma f(w) = f(\sigma(w))$.

Example 2. *For* σ: $1 \mapsto 1$, $2 \mapsto 21$, $3 \mapsto 32$, $\sigma(12312) = 1 \cdot 21 \cdot 32 \cdot 1 \cdot 21 = 12132121$.

$$\mathbf{M}_\sigma = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{M}_\sigma f(w) = f(\sigma(w)) = \begin{pmatrix} 4 \\ 3 \\ 1 \end{pmatrix}.$$

For two substitutions σ', σ'', we denote by $\circ$ their composition: $(\sigma'' \circ \sigma')(w) := \sigma''(\sigma'(w))$. Note that the above commutation relation implies that $\mathbf{M}_{\sigma'' \circ \sigma'} = \mathbf{M}_{\sigma''} \mathbf{M}_{\sigma'}$.

A substitution is said to be *unimodular* if $\det(\mathbf{M}_\sigma) = \pm 1$. In that case, $\mathbf{M}_\sigma$ is invertible and there is a formula involving $\mathbf{M}_\sigma^{-1}$ for the $(d-1)$-extension of σ [1, Proposition 2.11]. From now on, we assume that all substitutions are unimodular. The $(d-1)$-extension of σ acting on $(d-1)$-dimensional faces is then defined as follows:

$$E^{d-1}(\sigma)(\mathbf{x}, i^*) := \bigcup_{\substack{s \mid \sigma(j) = p \cdot i \cdot s \\ j \in \mathcal{A}_d}} (\mathbf{M}_\sigma^{-1}(\mathbf{x} + f(s)), j^*). \tag{6}$$

The subscript means that the union is done over all pairs (j, s) such that $j \in \mathcal{A}_d$ and $\sigma(j) = p \cdot i \cdot s$, where i is a letter, whereas p and s are words.

Example 3. *Let us consider the substitution* σ: $1 \mapsto 1$, $2 \mapsto 12$, $3 \mapsto 1213$ *over* $\mathcal{A}_3$ *and let us consider the image by* $E^2(\sigma)$ *of* $(\mathbf{0}, 1^*)$. *Since the type of that face is* 1, *the union in (6) must be done over all pairs* (j, s) *such that* $j \in \mathcal{A}_3$ *and* $\sigma(j) = p \cdot 1 \cdot s$. *There are four such pairs:* $(1, \epsilon)$, $(2, 2)$, $(3, 213)$, $(3, 3)$. *Furthermore, since the origin of the face is* $\mathbf{0}$, *every pair* (j, s) *gives the face* $(\mathbf{M}_\sigma^{-1}(f(s)), j^*)$. *Let us consider the third pair as an example:* $(3, 213)$ *corresponds to the face* $(\mathbf{M}_\sigma^{-1}(f(213)), 3^*)$, *where* $f(213) = \mathbf{e}_1 + \mathbf{e}_2 + \mathbf{e}_3$. *The image of* $(\mathbf{0}, 1^*)$ *thus consists of the following four faces:* $(\mathbf{0}, 1^*)$, $(\mathbf{M}_\sigma^{-1}\mathbf{e}_2, 2^*)$, $(\mathbf{M}_\sigma^{-1}(\mathbf{e}_1 + \mathbf{e}_2 + \mathbf{e}_3), 3^*)$, $(\mathbf{M}_\sigma^{-1}(\mathbf{e}_3), 3^*)$.

Geometrically, the action of $E^{d-1}(\sigma)$ can be considered as a *digitization* of the action of $\mathbf{M}_\sigma^{-1}$ (see Section 2 and Fig. 3 in [12]). Example 3 is illustrated in Fig. 7 with that point of view.

We extend (6) to unions of faces:

$$E^{d-1}(\sigma)(\mathcal{F}) := \bigcup_{(\mathbf{x}, i^*) \in \mathcal{F}} E^{d-1}(\sigma)(\mathbf{x}, i^*). \tag{7}$$

It is convenient to use the following notation for translations of faces: if $(\mathbf{x}, i^*)$ is a face and $\mathbf{y}$ is a vector, then $(\mathbf{x}, i^*) + \mathbf{y} := (\mathbf{x} + \mathbf{y}, i^*)$, which extends in a natural way to union of faces. One can then check that

$$E^{d-1}(\sigma)(\mathbf{x}, i^*) = \mathbf{M}_\sigma^{-1}\mathbf{x} + E^{d-1}(\sigma)(\mathbf{0}, i^*), \tag{8}$$

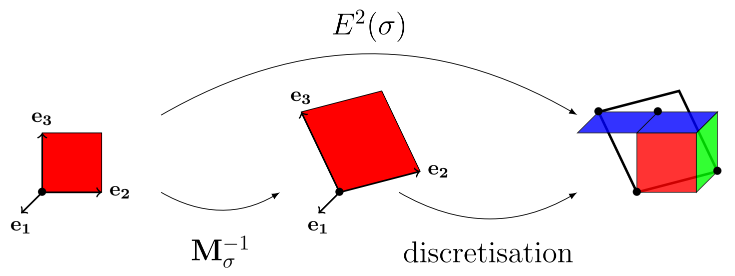


Figure 7: Image of $(\mathbf{0}, 1^*)$ by $\mathbf{M}_\sigma^{-1}$ and $E^2(\sigma)$, where σ is defined as in Example 3. The faces of type $1, 2, 3$ are respectively colored with red, green and blue.

because $\mathbf{M}_\sigma^{-1}\mathbf{x}$ does not depend on the union in (6). The images by $E^{d-1}(\sigma)$ of the faces $(\mathbf{0}, i^*)$ for $1 \leq i \leq d$ thus suffice to define $E^{d-1}(\sigma)$.

For two substitutions σ', σ'', we also denote by $\circ$ the composition of their extensions: $\big(E^{d-1}(\sigma'')\circ E^{d-1}(\sigma')\big)(\mathcal{F}) := E^{d-1}(\sigma'')\big(E^{d-1}(\sigma')(\mathcal{F})\big)$. We have the following key property [15, Proposition 1.2.4, item (1)]:

$$E^{d-1}(\sigma'') \circ E^{d-1}(\sigma') = E^{d-1}(\sigma' \circ \sigma''). \tag{9}$$

From (6), (7), (8) and (9), one can derive the following proposition, which shows that one can incrementally compute the image by $E^{d-1}(\sigma' \circ \sigma'')$ of a face $(\mathbf{0}, i^*)$ as a union-translation scheme:

Proposition 8. *We have for all $i \in \mathcal{A}_d$,*

$$E^{d-1}(\sigma' \circ \sigma'')(\mathbf{0}, i^*) = \bigcup_{\substack{s \mid \sigma'(j) = pis \\ j \in \mathcal{A}_d}} \big((\mathbf{M}_{\sigma' \circ \sigma''})^{-1} f(s) + E^{d-1}(\sigma'')(\mathbf{0}, j^*)\big).$$

Proof.

$$\begin{aligned} E^{d-1}(\sigma' \circ \sigma'')(\mathbf{0}, i^*) &= E^{d-1}(\sigma'')\big(E^{d-1}(\sigma')(\mathbf{0}, i^*)\big) \\ &= E^{d-1}(\sigma'')\Bigg(\bigcup_{\sigma'(j)=pis} (\mathbf{M}_{\sigma'}^{-1} f(s), j^*)\Bigg) \\ &= \bigcup_{\sigma'(j)=pis} E^{d-1}(\sigma'')\big(\mathbf{M}_{\sigma'}^{-1} f(s), j^*)\big) \\ &= \bigcup_{\sigma'(j)=pis} \big((\mathbf{M}_{\sigma''})^{-1}(\mathbf{M}_{\sigma'})^{-1} f(s) + E^{d-1}(\sigma'')(\mathbf{0}, j^*)\big) \\ &= \bigcup_{\sigma'(j)=pis} \big((\mathbf{M}_{\sigma' \circ \sigma''})^{-1} f(s) + E^{d-1}(\sigma'')(\mathbf{0}, j^*)\big). \end{aligned}$$

□

Example 4. *Let us consider the following substitutions:*

$$\sigma' : \begin{array}{l} 1 \mapsto 1 \\ 2 \mapsto 2 \\ 3 \mapsto 31 \end{array}, \quad \sigma'' : \begin{array}{l} 1 \mapsto 1 \\ 2 \mapsto 21 \\ 3 \mapsto 321 \end{array}, \quad \sigma' \circ \sigma'' : \begin{array}{l} 1 \mapsto 1 \\ 2 \mapsto 21 \\ 3 \mapsto 3121 \end{array}.$$

As shown in Fig. 8, σ' describes how the images by $E^2(\sigma' \circ \sigma'')$ relate to the images by $E^2(\sigma'')$. First, let us consider $E^2(\sigma' \circ \sigma'')(\mathbf{0}, 1^*)$, which is red in (b). It has been obtained as the union of two previous sets: $E^2(\sigma'')(\mathbf{0}, 1^*)$, which is red in (a), and $E^2(\sigma'')(\mathbf{0}, 3^*)$, which is blue in (a), because letter 1 belongs to both $\sigma'(1)$ and $\sigma'(3)$ and is also in the last position, which means with *no suffixe and thus no translation. Let us now consider $E^2(\sigma' \circ \sigma'')(\mathbf{0}, 3^*)$, which is blue in (b). It is a copy of $E^2(\sigma'')(\mathbf{0}, 3^*)$, which is blue in (a), translated by $\mathbf{M}_{\sigma'\circ\sigma''}\mathbf{e}_1 = \mathbf{e}_1$, because letter 3 appears only once in $\sigma'(3)$, followed by the single-letter word 1.*

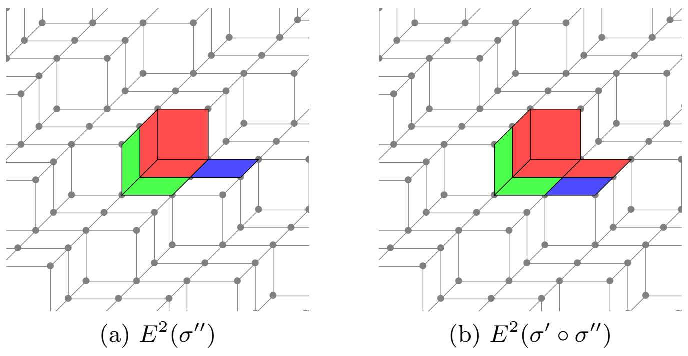

(a) $E^2(\sigma'')$ (b) $E^2(\sigma' \circ \sigma'')$

Figure 8: The images of $(\mathbf{0}, 1^*)$, $(\mathbf{0}, 2^*)$ and $(\mathbf{0}, 3^*)$ by $E^2(\sigma'')$ (a) and $E^2(\sigma'\circ\sigma'')$ (b) are displayed respectively in red, green and blue (see Example 4).

3.3 Sequences of Substitutions and Patterns

A sequence of matrices $(\mathbf{U}_n)_{0\leq n\leq N}$ that reduces a vector $\mathbf{v}$ is introduced in Definition 1. In this subsection, we define its *associated* sequence of substitutions.

Definition 5 (Associated substitutions)**.** *Let $(\mathbf{U}_n)_{0\leq n\leq N}$ be a sequence of matrices that reduces a vector $\mathbf{v} \in \mathbb{P}^d$. Its associated sequence of substitutions $(\sigma_n)_{0\leq n\leq N}$ verifies, for all $n \in \{0, \dots, N\}$, $\mathbf{M}_{\sigma_n} = ({}^{\mathrm{t}}\mathbf{U}_n)^{-1}$ and for all $i \in \mathcal{A}_d$, $\sigma_n(i)$ starts with letter i.*

Example 5. *Let us consider the following matrices:*

$$\mathbf{U}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix}, \quad ({}^{\mathrm{t}}\mathbf{U}_1)^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}.$$

Remark that $({}^{\mathrm{t}}\mathbf{U}_1)^{-1}$ is the incidence matrix of two disinct substitutions:

$$\sigma' : 1 \mapsto 1, 2 \mapsto 23, 3 \mapsto 3, \quad \sigma'' : 1 \mapsto 1, 2 \mapsto 32, 3 \mapsto 3.$$

Actually, the inverse of the transpose of all matrices in $\mathbb{U}^d$ is the incidence matrix of two distinct substitutions. According to Definition 5, $\sigma_1 := \sigma'$ because $\sigma'(i)$ starts with letter i for all $i \in \mathcal{A}_d$, which is not the case for σ''.

To save space, we set $\sigma_{n\cdots 0} := \sigma_n \circ \cdots \circ \sigma_0$ for $1 \leq n \leq N$. In accordance with Definition 1, we have thus:

$$\forall n \in \{0, \dots, N\}, \quad \mathbf{v}_n = ({}^{\mathrm{t}}\mathbf{M}_{\sigma_{n\cdots 0}})^{-1}\mathbf{v}, \quad \mathbf{a}_n = {}^{\mathrm{t}}\mathbf{M}_{\sigma_{n\cdots 0}}\mathbf{v}_N. \tag{10}$$

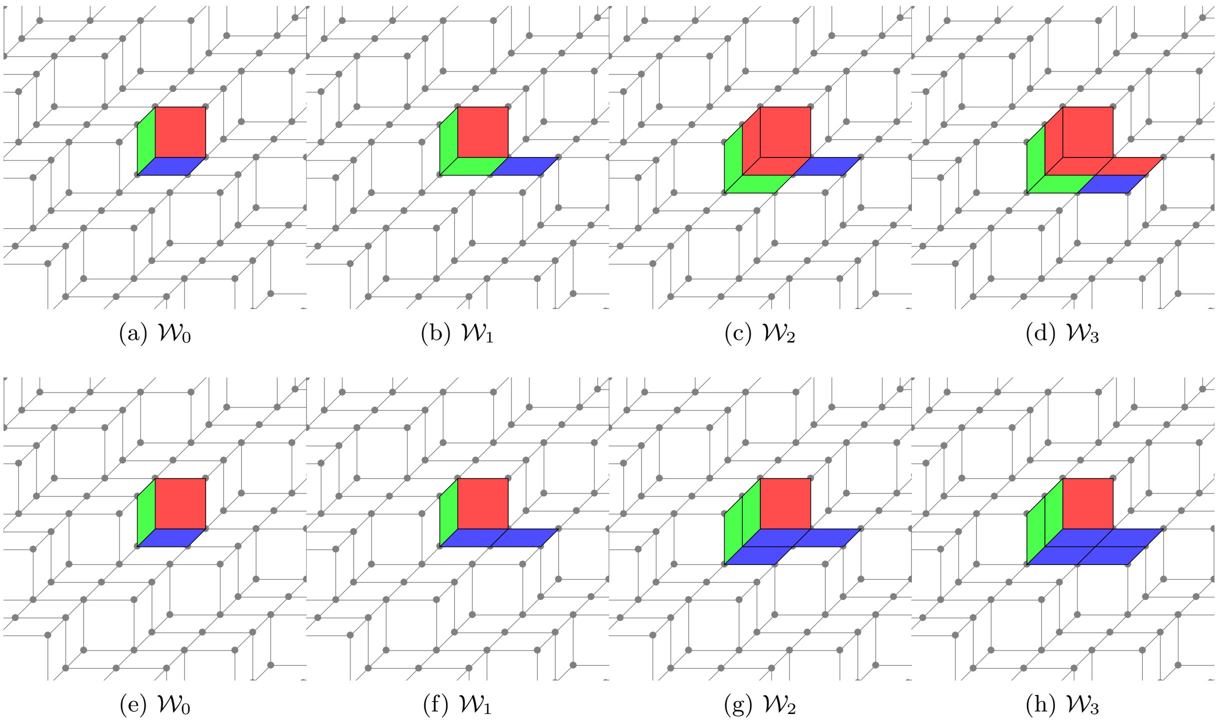
Table 2 shows, for a given sequence, how σ_n, $\mathbf{M}_n$, $\sigma_{n\cdots 0}$, $\mathbf{M}_{\sigma_{n\cdots 0}}$, $\mathbf{v}_n$ and $\mathbf{a}_n$ are related.

We will now focus on the image by $E^{d-1}(\sigma_{n\cdots 0})$ of faces with origin at $\mathbf{0}$. The result is a finite set of faces that we call *pattern*. A full example is given by Table 2 and Fig. 9.

Definition 6 (Pattern)**.** *Let $(\mathbf{U}_n)_{0\leq n\leq N}$ be a sequence of matrices that reduces a vector $\mathbf{v} \in \mathbb{P}^d$ and let $(\sigma_n)_{0\leq n\leq N}$ be its associated sequence of substitutions. Let $\mathcal{W}_0$ be the faces of type i and origin $\mathbf{0}$ such that ${}^{\mathrm{t}}\mathbf{v}_N\mathbf{e}_i = 1$, i.e., $\cup_{i\in\mathcal{A}_d, {}^{\mathrm{t}}\mathbf{v}_N\mathbf{e}_i=1}(\mathbf{0}, i^*)$. For all $n \in \{1, \dots, N\}$, let $\mathcal{W}_n$ be the image of $\mathcal{W}_0$ by $E^{d-1}(\sigma_{n\cdots 0})$, i.e., $\mathcal{W}_n := E^{d-1}(\sigma_{n\cdots 0})(\mathcal{W}_0)$.*

n	0	1	2	3
	$1 \mapsto 1$	$1 \mapsto 1$	$1 \mapsto 1$	$1 \mapsto 1$
σ_n	$2 \mapsto 2$	$2 \mapsto 2$	$2 \mapsto 21$	$2 \mapsto 2$
	$3 \mapsto 3$	$3 \mapsto 32$	$3 \mapsto 3$	$3 \mapsto 31$
$\mathbf{M}_n$	$\mathbf{I}_3$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
	$1 \mapsto 1$	$1 \mapsto 1$	$1 \mapsto 1$	$1 \mapsto 1$
$\sigma_{n\cdots 0}$	$2 \mapsto 2$	$2 \mapsto 2$	$2 \mapsto 21$	$2 \mapsto 21$
	$3 \mapsto 3$	$3 \mapsto 32$	$3 \mapsto 321$	$3 \mapsto 3121$
$\mathbf{M}_{\sigma_{n\cdots 0}}$	$\mathbf{I}_3$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$
${}^{\mathrm{t}}\mathbf{a}_n$	$\mathbf{1}$	$\begin{pmatrix} 1 & 1 & 2 \end{pmatrix}$	$\begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$	$\begin{pmatrix} 1 & 2 & 4 \end{pmatrix}$
${}^{\mathrm{t}}\mathbf{v}_n$	$\begin{pmatrix} 1 & 2 & 4 \end{pmatrix}$	$\begin{pmatrix} 1 & 2 & 2 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 2 \end{pmatrix}$	$\mathbf{1}$

Table 2: A sequence of substitutions and other related quantities.


(a) $\mathcal{W}_0$ (b) $\mathcal{W}_1$ (c) $\mathcal{W}_2$ (d) $\mathcal{W}_3$

(e) $\mathcal{W}_0$ (f) $\mathcal{W}_1$ (g) $\mathcal{W}_2$ (h) $\mathcal{W}_3$

Figure 9: The patterns are the images by $E^2(\sigma_{n\cdots 0})$ of the faces with origin at $\mathbf{0}$. The sequence of substitutions is given in Table 2. It is associated with a sequence of matrices obtained from the reduction of $(1, 2, 4)$ using algorithm H [20]. In the top row, the images of $(\mathbf{0}, 1^*)$, $(\mathbf{0}, 2^*)$ and $(\mathbf{0}, 3^*)$ by $E^2(\sigma_{n\cdots 0})$ are displayed respectively in red, green and blue. In the bottom row, the faces of type $1, 2, 3$ are respectively colored with red, green and blue.

From a sequence of substitutions $(\sigma_n)_{0\leq n\leq N}$, there are at least two ways of generating patterns. In the first method, we compute the substitution $\sigma_{n\cdots 0}$ and then apply the formula given by (6) to get $\mathcal{W}_n$. The second method is incremental: we compute $\mathcal{W}_n$ from $\mathcal{W}_{n-1}$ as in a union-translation scheme thanks to Proposition 8. One can check that the time-complexity of both methods linearly depends on the number of faces in $\mathcal{W}_n$. By Proposition 10 or Theorem 11, item (iii), proven in the next subsection, this number is equal to $\|\mathbf{a}_n\|_1$. As a result, the overall time-complexity of both methods is $O(\|\mathbf{a}_n\|_1)$.

From a vector $\mathbf{v}$, one has to first compute a sequence of matrices that reduces $\mathbf{v}$ and then compute the associated sequence of substitutions. The overall time-complexity thus depends on the reduction algorithm. For most of them, e.g., Brun, Selmer, H [20], every matrix choice takes a constant time, while there are at most $O(\|\mathbf{v}\|_1)$ choices in total (Theorem 3). In that case, the reduction stage takes $O(\|\mathbf{v}\|_1)$ time. Then, the generation of $\mathcal{W}_N$ also takes $O(\|\mathbf{v}\|_1)$ time, because $\mathbf{a}_N = \mathbf{v}$ by (1). However, note that there are reduction algorithms, e.g., R [20], L [21], which does not compute new matrices in constant time, but in logarithmic time, thus increasing the whole time-complexity by a logarithmic factor.

3.4 Properties of Patterns

In this subsection, we show several properties for the patterns defined in Definition 6. Even if a pattern is a set of faces, we will say that a point $\mathbf{x}$ is in a pattern $\mathcal{W}$, denoted by $\mathbf{x} \in \mathcal{W}$ by abuse of notation, whenever there exists a face $w \in \mathcal{W}$ such that $\mathbf{x}$ is a vertex of w.

The following proposition shows that the patterns always contain some very specific faces and points.

Proposition 9. *Let us assume that* $\mathbf{v}_N \in \{\mathbf{1}\} \cup \{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$ *and let* $\mathcal{A}_{\mathbf{v}_N}$ *be equal to* $\{i \in \mathcal{A}_d \mid {}^{\mathrm{t}}\mathbf{v}_N\mathbf{e}_i = 1\}$. *For all* $n \in \{0, \ldots, N\}$ *and for all* $i \in \mathcal{A}_{\mathbf{v}_N}$, $(\mathbf{e}_i - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\mathbf{e}_i, i^*) \in \mathcal{W}_n$ *and* $\mathbf{1} - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\mathbf{e}_i \in \mathcal{W}_n$.

Proof. Since every word $\sigma_0(i), \ldots, \sigma_n(i)$ starts with i, $\sigma_{n\cdots 0}(i)$ also starts with i. Let s be the suffix such that $\sigma_{n\cdots 0}(i) = i \cdot s$. The face $(\mathbf{M}^{-1}_{\sigma_{n\cdots 0}}f(s), i^*)$ is by definition included in $E^d(\sigma_{n\cdots 0})(\mathbf{0}, i^*)$ and thus in $\mathcal{W}_n$ for all $i \in \mathcal{A}_{\mathbf{v}_N}$.

In that case, $f(s) = f(\sigma_{n\cdots 0}(i)) - \mathbf{e}_i$, where $f(\sigma_{n\cdots 0}(i))$ is the i-th column of $\mathbf{M}_{\sigma_{n\cdots 0}}$ and is thus equal to $\mathbf{M}_{\sigma_{n\cdots 0}}\mathbf{e}_i$. We have therefore:

$$\begin{aligned}\mathcal{W}_n &\ni (\mathbf{M}^{-1}_{\sigma_{n\cdots 0}}f(s), i^*)\\ &= (\mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\big(f(\sigma_{n\cdots 0}(i)) - \mathbf{e}_i\big), i^*)\\ &= (\mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\big(\mathbf{M}_{\sigma_{n\cdots 0}}\mathbf{e}_i - \mathbf{e}_i\big), i^*)\\ &= (\mathbf{e}_i - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\mathbf{e}_i, i^*).\end{aligned}$$

In addition, the point $\mathbf{e}_i + \sum_{k\neq i}\mathbf{e}_k - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\mathbf{e}_i = \mathbf{1} - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\mathbf{e}_i$ is a vertex of $(\mathbf{e}_i - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\mathbf{e}_i, i^*)$ by (4) and thus is in $\mathcal{W}_n$. □

In plane-probing algorithms, the point used to select a new matrix at a step n is exactly $\mathbf{1} - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}}\mathbf{e}_i$ (see Proposition 6 and (3)) and thus lies in $\mathcal{W}_n$ by Proposition 9. The convention used in Definition 5 in order to associate only one substitution to a given matrix turns out to guarantee that the generated patterns contain those points.

The following proposition shows that all faces in $\mathcal{W}_n$ have a specific location in space with respect to direction $\mathbf{a}_n$. It is key for several other properties stated in Theorem 11.

Proposition 10. *Let us assume that* $\mathbf{v}_N \in \{\mathbf{1}\} \cup \{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$. *For all* $n \in \{0, \ldots, N\}$ *and for all* $j \in \mathcal{A}_d$, *for every face* $(\mathbf{x}, j^*) \in \mathcal{W}_n$, *the quantity* ${}^{\mathrm{t}}\mathbf{a}_n\mathbf{x}$ *appears once and only once in the integer interval* $\{0, \ldots, {}^{\mathrm{t}}\mathbf{a}_n\mathbf{e}_j - 1\}$.

Proof. By Definition 6 and according to (6), all faces of type j in $\mathcal{W}_n$ are located along a path encoded by $\sigma_{n\cdots 0}(j)$.

If there is only one letter i in $\sigma_{n\cdots 0}(j)$, there is only one face in $E^{d-1}(\sigma_{n\cdots 0})(\mathbf{0}, i^*)$ whose origin is $\mathbf{0}$ and the proposition is obviously true.

Otherwise, let us denote by a and b two consecutive letters in $\sigma_{n\cdots 0}(j)$. In other words $\sigma_{n\cdots 0}(j) = p \cdot a \cdot b \cdot s$. Letter a (resp. b) corresponds to a face in $E^{d-1}(\sigma_{n\cdots 0})(\mathbf{0}, a^*)$ (resp. $E^{d-1}(\sigma_{n\cdots 0})(\mathbf{0}, b^*)$). By abuse of terminology, we call the faces a and b and say that they are consecutive. The difference between the origins is equal to $\mathbf{M}^{-1}_{\sigma_{n\cdots 0}} f(bs) - \mathbf{M}^{-1}_{\sigma_{n\cdots 0}} f(s) = \mathbf{M}^{-1}_{\sigma_{n\cdots 0}} \mathbf{e}_b$. However, the quantity ${}^{\mathrm{t}}\mathbf{a}_n \mathbf{M}^{-1}_{\sigma_{n\cdots 0}} \mathbf{e}_b$ is equal to

$$\left({}^{\mathrm{t}}\mathbf{v}_N \mathbf{M}_{\sigma_{n\cdots 0}} \right) \mathbf{M}^{-1}_{\sigma_{n\cdots 0}} \mathbf{e}_b = {}^{\mathrm{t}}\mathbf{v}_N \mathbf{e}_b.$$

Remind that if ${}^{\mathrm{t}}\mathbf{v}_N \mathbf{e}_b = 1$, the face of $(\mathbf{M}^{-1}_{\sigma_{n\cdots 0}} \mathbf{e}_b, j^*)$ is kept in $\mathcal{W}_n$. Otherwise, ${}^{\mathrm{t}}\mathbf{v}_N \mathbf{e}_b = 0$ and the face is discarded. As a consequence, the dot product between $\mathbf{a}_n$ and the difference between the origins of a and b is 1 if b is in $\mathcal{W}_n$ and 0 otherwise. The set $\{ {}^{\mathrm{t}}\mathbf{a}_n \mathbf{x} \mid (\mathbf{x}, j^*) \in \mathcal{W}_n \}$ thus form an integer interval whose lower bound is 0. The upper bound plus one is given by the total number of faces of type j in $\mathcal{W}_n$, which is equal to the number of letters $i \in \sigma_{n\cdots 0}(j)$ such that ${}^{\mathrm{t}}\mathbf{v}_N \mathbf{e}_i = 1$:

$${}^{\mathrm{t}}\mathbf{v}_N f(\sigma_{n\cdots 0}(j)) = {}^{\mathrm{t}}\mathbf{v}_N \mathbf{M}_{\sigma_{n\cdots 0}} \mathbf{e}_j = {}^{\mathrm{t}}\mathbf{a}_n \mathbf{e}_j.$$

□

We are now able to gather several properties of patterns into our main theorem:

Theorem 11. *Let us assume that $\mathbf{v}_N \in \{\mathbf{1}\} \cup \{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$. The three following properties hold on the patterns defined in Definition 6:*

(i) $\forall n \in \{0, \ldots, N\}$, *$\mathcal{W}_n \subset \mathcal{S}_{\mathbf{a}_n}$ and $\mathcal{W}_n \subset \mathcal{S}_{\mathbf{a}_N}$.*

(ii) $\forall n \in \{0, \ldots, N\}$, *the copies of $\mathcal{W}_n$ translated along the lattice Λ_n (Definition 2 and Proposition 5) is a partition of $\mathcal{S}_{\mathbf{a}_n}$.*

(iii) $\forall n \in \{0, \ldots, N\}$, $\forall j \in \mathcal{A}_d$, *$\mathcal{W}_n$ has ${}^{\mathrm{t}}\mathbf{a}_n \mathbf{e}_j$ faces of type j.*

Two extra properties hold if $\mathbf{v}_N = \mathbf{1}$:

(iv) $\forall n \in \{1, \ldots, N\}$, $\mathcal{W}_{n-1} \subset \mathcal{W}_n$,

(v) $\forall n \in \{0, \ldots, N\}$, *$\mathcal{W}_n$ contains points from which one can compute the vectors of a basis of Λ_n.*

Proof. (i) By Proposition 10, if $\mathcal{W}_n$ contains at least one face of type $j \in \mathcal{A}_d$, then, for every face $(\mathbf{x}, j^*) \in \mathcal{W}_n$, we have $0 \leq {}^{\mathrm{t}}\mathbf{a}_n \mathbf{x} < {}^{\mathrm{t}}\mathbf{a}_n \mathbf{e}_j$, which means that $(\mathbf{x}, j^*)$ is in the stepped plane of normal $\mathbf{a}_n$ according to (5).

In addition, if $n < N$, note that the coordinates of $\mathbf{a}_n$ are always nonnegative and the multiplication by ${}^{\mathrm{t}}\mathbf{M}_{\sigma_n}$ consists in adding a coordinate to another, so that $\mathbf{a}_n \leq \mathbf{a}_{n+1}$. By induction, we get $\mathbf{a}_n \leq \mathbf{a}_N$ and in particular ${}^{\mathrm{t}}\mathbf{a}_n \mathbf{e}_j \leq {}^{\mathrm{t}}\mathbf{a}_N \mathbf{e}_j$. As a consequence, all faces of $\mathcal{W}_n$ are also in the stepped plane of normal $\mathbf{a}_N$.

(ii) By Proposition 10, if $\mathcal{W}_n$ contains at least one face of type $j \in \mathcal{A}_d$, then, for every face $(\mathbf{x}, j^*) \in \mathcal{W}_n$, the quantity ${}^{\mathrm{t}}\mathbf{a}_n \mathbf{x}$ appears once and only once in the integer interval $\{0, \ldots, {}^{\mathrm{t}}\mathbf{a}_n \mathbf{e}_j - 1\}$. Since the copies of $\mathcal{W}_n$ are translated along the lattice Λ_n, i.e., by a translation vector whose dot product with $\mathbf{a}_n$ is zero, the union of all copies is the set $\{(\mathbf{x}, j^*) \mid 0 \leq {}^{\mathrm{t}}\mathbf{a}_n \mathbf{x} < {}^{\mathrm{t}}\mathbf{a}_n \mathbf{e}_j\}$. Taking into account all face types, we get the stepped plane of normal $\mathbf{a}_n$ according to (5).

It remains to prove that two distinct copies of $\mathcal{W}_n$ cannot share a common face $(\mathbf{x}, i^*)$. Let us denote by $\mathcal{W}'$ and $\mathcal{W}''$ the two copies and by $\mathbf{t} \neq \mathbf{0}$ the translation vector such that $\mathcal{W}'' = \mathcal{W}' + \mathbf{t}$. If $(\mathbf{x}, i^*) \in \mathcal{W}''$, then $(\mathbf{x} - \mathbf{t}, i^*) \in \mathcal{W}'$. By definition of Λ_n, we have ${}^{\mathrm{t}}\mathbf{a}_n \mathbf{t} = 0$, which implies that

${}^{\mathrm{t}}\mathbf{a}_n\mathbf{x} = {}^{\mathrm{t}}\mathbf{a}_n(\mathbf{x}-\mathbf{t})$. However, by Proposition 10, there are no two faces in $\mathcal{W}_n$ whose origin has the same dot product with ${}^{\mathrm{t}}\mathbf{a}_n$. Since that fact is obviously translation invariant, the same is true for $\mathcal{W}'$, which means that $(\mathbf{x}, i^*)$ and $(\mathbf{x}-\mathbf{t}, i^*)$ cannot be both in $\mathcal{W}'$. Consequently, $(\mathbf{x}, i^*)$ cannot belong to both $\mathcal{W}'$ and $\mathcal{W}''$.

(iii) Direct consequence of Proposition 10.

(iv) if $\mathbf{v}_N = \mathbf{1}$, we have for all $n \in \{1, \ldots, N\}$:

$$\begin{array}{lll} \mathcal{W}_n & = \cup_{i\in\mathcal{A}_d} E^{d-1}(\sigma_{n\cdots 0})(\mathbf{0}, i^*) & \text{(Definition 6)} \\ & = \cup_{i\in\mathcal{A}_d} \cup_{\sigma_n(j)=pis} \left((\mathbf{M}_{\sigma_n\cdot 0})^{-1} f(s) + E^{d-1}(\sigma_{n-1\cdots 0})(\mathbf{0}, j^*)\right) & \text{(Proposition 8)} \\ & \supset \cup_{j\in\mathcal{A}_d} E^{d-1}(\sigma_{n-1\cdots 0})(\mathbf{0}, j^*) = \mathcal{W}_{n-1} & \text{(Definition 6)}, \end{array}$$

where the inclusion comes from the trivial fact that for all $j \in \mathcal{A}_d$, $\sigma_n(j)$ may always be written $p \cdot i \cdot \epsilon$, where i is any letter in $\mathcal{A}_d$, i.e., the word $\sigma_n(j)$ ends with a letter $i \in \mathcal{A}_d$.

(v) By Proposition 5, the vectors of a basis of Λ_n are

$$\left(\mathbf{M}_{\sigma_n\cdots 0}^{-1}(\mathbf{e}_i - \mathbf{e}_{i-1})\right)_{i\in\{2,\ldots,d\}}.$$

In addition, by Proposition 9, $\mathcal{W}_n$ contains the points

$$\left(\mathbf{1} - \mathbf{M}_{\sigma_n\cdots 0}^{-1}\mathbf{e}_i\right)_{i\in\{1,\ldots,d\}}.$$

Consequently, for all $i \in \{2, \ldots, d\}$, we have:

$$\mathbf{1} - \mathbf{M}_{\sigma_n\cdots 0}^{-1}\mathbf{e}_{i-1} - (\mathbf{1} - \mathbf{M}_{\sigma_n\cdots 0}^{-1}\mathbf{e}_i) = \mathbf{M}_{\sigma_n\cdots 0}^{-1}(\mathbf{e}_i - \mathbf{e}_{i-1}),$$

which concludes the proof. □

Theorem 11 shows that our method provides a sequence of patterns, all included in a given stepped plane (i). The last pattern periodically generates the underlying stepped plane without hole or overlap (ii) and is of minimal size (iii) because if one sum the normal vector of all its faces, we get exactly $\mathbf{a}_N$, i.e., the normal of the stepped plane, and one cannot expect to find a smaller pattern with the same normal.

In addition, if $\mathbf{v}_N = \mathbf{1}$, the patterns form a hierarchical set by inclusion (iv) and they contain points from which one can compute the period vectors (v).

3.5 Geometrical and Topological Remarks

In this subsection, we discuss two additional and desirable properties: shape compactness and connectivity.

Our first remark is that the choice of the algorithm has a great impact on the shape of the pattern (see Fig. 10).

Experiments were conducted to measure how far the patterns are from the origin. We generated patterns using the reduction algorithms H, Brun (B), Selmer (S) and Jacobi-Perron (JP) on 811214 vectors with relatively prime components ranging from $(1, 1, 1)$ to $(100, 100, 100)$. We discarded R and L, which behave almost exactly like H in that range, as well as Poincaré, which probes for points very far away from the origin as shown in Fig. 5. For all input vectors, we computed the maximal distance between every point of the pattern and the origin. In Fig. 11, we plotted that maximal distance against the magnitude of the input vector. We smoothed the lines as in Fig. 5. The patterns generated from the geometrical approach (H) are on average much closer to the origin than those generated from the arithmetical approach (B, S and JP).

In our opinion, this is because the basis of the lattice Λ_n returned by the algorithm R and L (resp. H) is always (resp. almost always) reduced [20, 24]. There are no such results with classical three-dimensional Euclidean algorithms, because they are not geometry-aware. A perspective is to bound the distance of the pattern boundary to the origin. For the algorithms

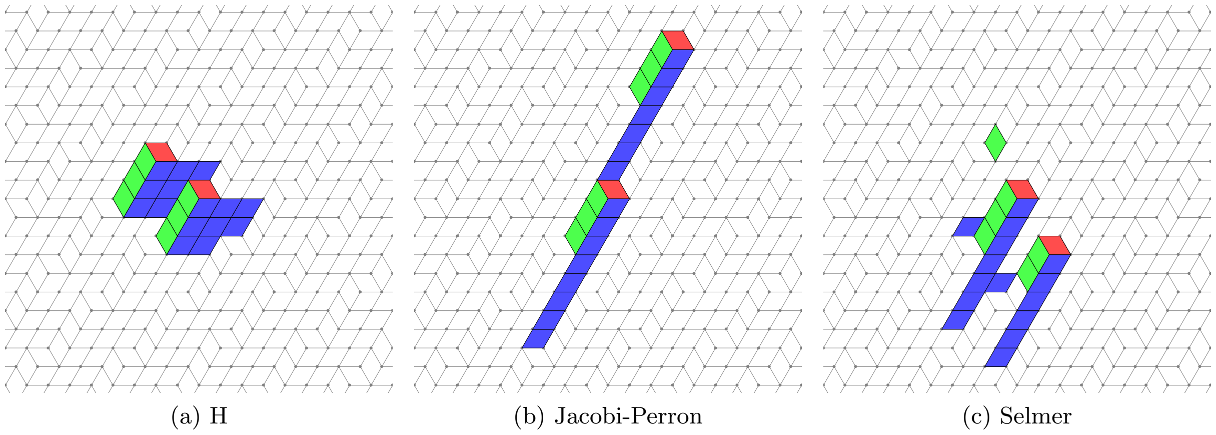

(a) H (b) Jacobi-Perron (c) Selmer

Figure 10: Patterns of normal ${}^{t}\mathbf{v} = (2, 6, 15)$ generated by H (a), Jacobi-Perron (b) and Selmer (c). In (a) and (b), the patterns are edge-connected, whereas in (c), there are two vertex-connected components and three edge-connected components. Faces of type 1, 2 and 3 are respectively displayed in red, green and blue.

H, R and L, such bound may involve geometrical arguments based on the empty-circumsphere criterion.

Our second remark is about the connectivity of the patterns. In general, the patterns may be neither vertex- nor edge-connected. For instance, with Selmer, an example of non-connected pattern is given in Fig. 10 (c). On the contrary, with Brun and Jacobi-Perron, the patterns are always edge-connected as proven in [6, 15]. Note that the tie-break rule mentioned in Subsection 2.5 is important to guarantee connectivity.

For all vectors with relatively prime components ranging from $(1, 1, 1)$ to $(100, 100, 100)$, we checked whether the pattern is edge-connected or not. Results are reported in Table 3.

Algorithm	H	S	B	JP
Edge-connected (%)	73.11	43.04	100	100

Table 3: Patterns were generated from the reduction algorithms H, Selmer (S), Brun (B) and Jacobi-Perron (JP) on all vectors ranging from $(1, 1, 1)$ to $(100, 100, 100)$ with relatively prime coordinates.

Unfortunately, there is an opposition between proximity and connectivity. Indeed, Brun and Jacobi-Perron provide patterns that are edge-connected, but also much less close to the origin than the patterns obtained from the geometrical approach, which may be not connected however. It seems that one cannot have all geometrical, topological and combinatorial properties at the same time, but that statement has yet to be rigorously proven.

The lack of guarantee of connectivity is clearly a drawback. It seems indeed natural to require that patterns must be connected. However, even unconnected patterns may be good candidate to locally approximate tangent planes in the context of digital surface analysis. We tend to think that shape compactness is a more determinant property than connectivity for an accurate and convergent approximation of tangent plane but this has to be further studied.

To end the section, let us mention the method developed in [14], which generates, from an input vector $\mathbf{v}$, a set of points included in a digital plane of normal $\mathbf{v}$. It is incremental and based on several three-dimensional continued-fraction algorithms. It generates point sets that are provably connected but that may have holes. In addition, those sets are not thick enough, e.g., they do not contain any point $\mathbf{x}$ such that ${}^{t}\mathbf{v}\mathbf{x} = \|\mathbf{v}\|_1 - 1$, and are composed of a great amount of points that cover the digital plane with overlaps. That is why we believe that our

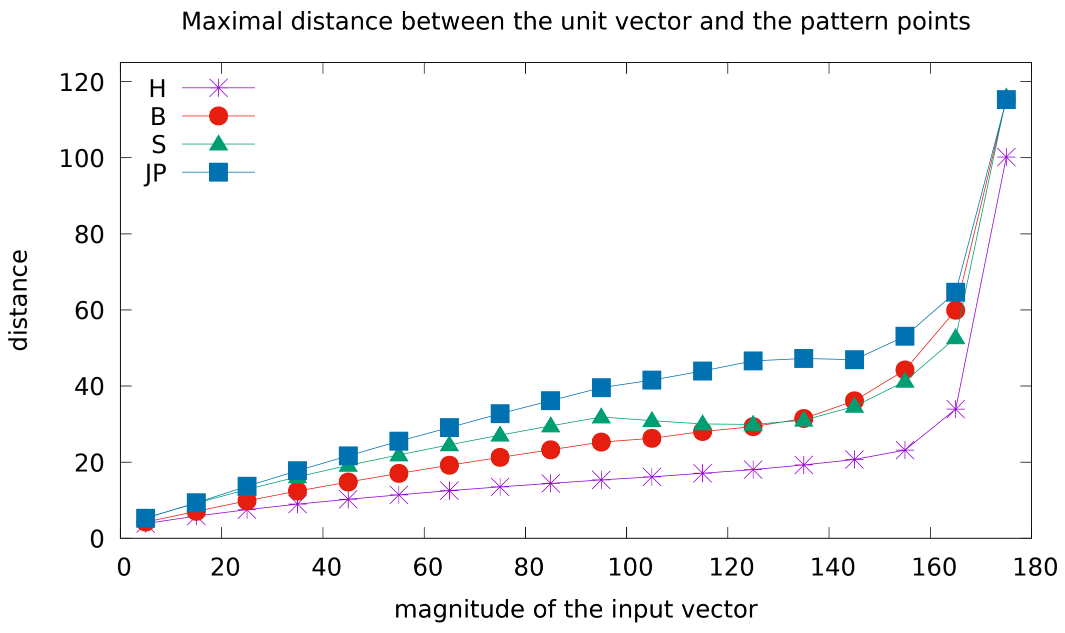


Figure 11: For all vectors with relatively prime components ranging from $(1,1,1)$ to $(100,100,100)$, the maximal distance between the origin and every point of the pattern is plotted against the magnitude of the vector.

method has a greater practical interest in the context of digital surface analysis.

4 Conclusion

We have introduced a d-dimensional generalization of the Euclidean algorithm, which covers a lot of algorithms: not only Euclid's algorithm itself and the classical three-dimensional Euclidean algorithms like Brun, Selmer, Poincaré or Jacobi-Perron, but also plane-probing algorithms recently developed in the context of digital geometry. All the classical three-dimensional Euclidean algorithms can even be translated into a plane-probing version due to Proposition 6 and the discussion that follows.

All those algorithms share in common that they return a sequence of elementary matrices that transforms a given normal vector into a terminal element. That output can be used together with a geometrical extension of substitutions, to generate sets of $(d-1)$-dimensional faces.

The generated sets have mainly three combinatorial properties listed in Theorem 11: they can form a hierarchical set of patterns, all included in a given digital hyperplane. The pattern of the highest level is of minimal size and periodically generates the underlying digital hyperplane without hole or overlap, hence the name pattern. In addition, those patterns experimentally have an additional geometrical property when used together with a plane-probing algorithm: they are rather compact and close to the origin as illustrated in Fig. 10. It is however a perspective to quantify compactness and proximity and to provide upper bounds on such measures. To the best of our knowledge, no other method, whether it was based on digitization or based on union-translation schemes, has all the above properties.

Plane-probing algorithms compute a normal vector by iteratively updating a triangle that locally fits a digital plane or, with slight changes, a digital surface. If the digital surface is not convex, the triangles may jump over holes or stab the digital surface, because the set of probes, which is too sparse, does not faithfully represent the digital surface. In order to correctly approximate the tangent plane in such cases, our idea is to incrementally compute a pattern instead of a triangle: new probes provide the next matrix in the sequence, we then use the proposed method to generate the next pattern and we finally check whether it fits the digital surface or not, as illustrated in Fig. 1. We plan to thoroughly study how accurately do such patterns approximate tangent planes. We hope it will improve the local analysis of digital surfaces using plane-probing.

References

[1] P. Arnoux, M. Furukado, E. Harriss, and S. Ito. Algebraic numbers, free group automorphisms and substitutions on the plane. *Trans. Amer. Math. Soc.*, 363:4651–4699, 2011.

[2] P. Arnoux and S. Ito. Pisot substitutions and Rauzy fractals. *Bulletin of the Belgian Mathematical Society Simon Stevin*, 8(2):181–208, 2001.

[3] V. Berthé, É. Domenjoud, D. Jamet, and X. Provençal. Fully Subtractive algorithm, tribonacci numeration and connectedness of discrete planes. *Research Institute for Mathematical Sciences, Lecture note Kokyuroku Bessatu B*, 46:159–174, 2014.

[4] V. Berthé and T. Fernique. Brun expansions of stepped surfaces. *Discret. Math.*, 311(7):521–543, 2011.

[5] V. Berthé, D. Jamet, T. Jolivet, and X. Provençal. Critical connectedness of thin arithmetical discrete planes. In *Proc. DGCI*, pages 107–118, 2013.

[6] V. Berthé, A. Lacasse, G. Paquin, and X. Provençal. A study of Jacobi-Perron boundary words for the generation of discrete planes. *Theor. Comput. Sci.*, 502:118 – 142, 2013.

[7] V. Brimkov, D. Coeurjolly, and R. Klette. Digital planarity – a review. *Discret. Appl. Math.*, 155(4):468–495, 2007.

[8] E. Domenjoud, B. Laboureix, and L. Vuillon. Facet connectedness of arithmetic discrete hyperplanes with non-zero shift. In *Proc. DGCI*, 2019.

[9] E. Domenjoud, X. Provençal, and L. Vuillon. Facet connectedness of discrete hyperplanes with zero intercept: The general case. In *Proc. DGCI*, pages 1–12, 2014.

[10] E. Domenjoud and L. Vuillon. Geometric Palindromic Closure. *Uniform Distribution Theory*, 7(2):109–140, 2012.

[11] T. Fernique. Multidimensional Sturmian Sequences and Generalized Substitutions. *International Journal of Foundations of Computer Science*, 17:575–600, 2006.

[12] T. Fernique. Generation and recognition of digital planes using multi-dimensional continued fractions. *Pattern Recognition*, 42(10):2229–2238, 2009.

[13] J. Françon, J.-M. Schramm, and M. Tajine. Recognizing arithmetic straight lines and planes. In *Proc. DGCI*, pages 139–150, 1996.

[14] D. Jamet, N. Lafrenière, and X. Provençal. Generation of digital planes using generalized continued-fractions algorithms. In *Proc. DGCI*, pages 45–56, 2016.

[15] T. Jolivet. *Combinatorics of Pisot Substitutions.* Phd thesis, Université Paris Diderot, University of Turku, 2013.

[16] R. Klette and A. Rosenfeld. Digital straightness – a review. *Discret. Appl. Math.*, 139(1-3):197–230, 2004.

[17] S. Labbé. 3-dimensional continued fraction algorithms cheat sheets. *arXiv preprint arXiv:1511.08399*, 2015.

[18] S. Labbé and C. Reutenauer. A d-dimensional Extension of Christoffel Words. *Discrete and Computational Geometry*, 54(1):152–181, 2015.

[19] J.-O. Lachaud, J. Meyron, and T. Roussillon. An optimized framework for plane-probing algorithms. *J. Math. Imaging Vis.*, 62:718–736, 2020.

[20] J.-O. Lachaud, X. Provençal, and T. Roussillon. Two plane-probing algorithms for the computation of the normal vector to a digital plane. *J. Math. Imaging Vis.*, 59:23–39, 2017.

[21] J.-T. Lu, T. Roussillon, and D. Coeurjolly. A New Lattice-based Plane-probing Algorithm. In *IAPR Second International Conference on Discrete Geometry and Mathematical Morphology*, Strasbourg, France, Oct. 2022.

[22] J. Meyron and T. Roussillon. Approximation of Digital Surfaces by a Hierarchical Set of Planar Patches. In *IAPR Second International Conference on Discrete Geometry and Mathematical Morphology*, pages 409–421, Strasbourg, France, Oct. 2022.

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

[24] T. Roussillon, J.-T. Lu, J.-O. Lachaud, and D. Coeurjolly. Delaunay property and proximity results of the L-algorithm. Research report, Université de Lyon, July 2022.

[25] A. Troesch. Interprétation géométrique de l'algorithme d'Euclide et reconnaissance de segments. *Theor. Comput. Sci.*, 115(2):291 – 319, 1993.