Noname manuscript No.
(will be inserted by the editor)

Efficient Distance Transformation for Path-based Metrics

David Coeurjolly · **Isabelle Sivignon**

the date of receipt and acceptance should be inserted later

Abstract In many applications, separable algorithms have demonstrated their efficiency to perform high performance volumetric processing of shape, such as distance transformation or medial axis extraction. In the literature, several authors have discussed about conditions on the metric to be considered in a separable approach. In this article, we present generic separable algorithms to efficiently compute Voronoi maps and distance transformations for a large class of metrics. Focusing on path-based norms (chamfer masks, neighborhood sequences...), we propose efficient algorithms to compute such volumetric transformation in dimension n. We describe a new $O(n \cdot N^n \cdot \log N \cdot (n + \log f))$ algorithm for shapes in a N^n domain for chamfer norms with a rational ball of f facets (compared to $O(f^{\lfloor \frac{n}{2} \rfloor} \cdot N^n)$ with previous approaches). Last we further investigate an even more elaborate algorithm with the same worst-case complexity, but reaching a complexity of $O(n \cdot N^n \cdot \log f \cdot (n + \log f))$ experimentally, under assumption of regularity distribution of the mask vectors.

Keywords Digital Geometry, Distance Transformation, Path-based Norms

This work has been partially funded by COMEDIC ANR-15-CE40-0006 research grant.

David Coeurjolly
Université de Lyon, CNRS, LIRIS, UMR5205, F-69621, France
E-mail: david.coeurjolly@liris.cnrs.fr

Isabelle Sivignon
Univ. Grenoble Alpes, CNRS, Grenoble INP*, GIPSA-lab, 38000 Grenoble, France
E-mail: isabelle.sivignon@gipsa-lab.grenoble-inp.fr
*Institute of Engineering Univ. Grenoble Alpes

1 Introduction

Volumetric analysis of digital shapes is crucial in many geometry processing applications, for instance to be able to measure distances between two points in $\mathbb{Z}^n$, or to measure the width of a shape or the proximity between two shapes. Since early works on digital geometry, distance transformation has been widely investigated [26,27]. Given a finite input shape $X \subset \mathbb{Z}^n$, the distance transformation labels each point in X with the distance to its closest point in $\mathbb{Z}^n \setminus X$. Labeling each point by the closest background point leads to Voronoi maps (*e.g.* the restriction to $\mathbb{Z}^n$ of Voronoi diagrams from computational geometry [13]). Since such characterization is parametrized by a distance function, many authors have addressed this distance transformation problem with trade-offs between algorithmic performances and the *accuracy* of the digital distance function with respect to the Euclidean one. Hence, authors have considered distances based on chamfer masks [27,6,14] or sequences of chamfer masks [26,22,28,24]; the vector displacement based Euclidean distance [12,25]; Voronoi diagram based Euclidean distance [7,20] or the square of the Euclidean distance [15,21]. For the Euclidean metric, separable volumetric computations have demonstrated to be very efficient with the design of optimal $O(n \cdot N^n)$ time algorithms for shapes in N^n domains, optimal multithread/GPU implementation or extensions to toric domains (please refer to [9] for a discussion). For path-based metrics (*e.g.* chamfer mask, -weighted- neighborhood sequences), two main techniques exist to compute the distance transformation. The first one considers a weighted graph formulation of the problem and Dijkstra-like algorithms on weighted graphs to compute distances. If m denotes the size of the chamfer mask, computational cost could be in $O(m \cdot N^n)$ using a cyclic bucket data structure [30]. Another approach consists in a raster scan of the domain: first the chamfer mask is decomposed into disjoint sub-masks; then the

domain grid points are scanned in a given order (consistent with the sub-mask construction) and a local computation is performed before being propagated [26,6]. Scanning the domain several times (one per sub-mask) leads to the distance transformation values. Again, we end up with a $O(m \cdot N^n)$ computational cost. Besides specific applications which use the anisotropic nature of the chamfer mask, rotational dependency is usually enforced by increasing the mask size m (its number of vectors, see below) leading to expensive computational costs.

In [10], we have demonstrated that in dimension 2, we can adapt separable algorithms used for the Euclidean distance to extract path-based metric distance transformation in $O(\log^2 m \cdot N^2)$.

Contributions This article details the preliminary analysis of [10] and extend it to higher dimensional distance transformation problems. More precisely, we describe efficient and parallel algorithms in arbitrary dimension n to compute error-free distance transformation and Voronoi map for chamfer norms and other path-based metrics. Overall computational costs are summarized in Table 1 (see 3.2 for predicate definitions).

The article is organized as follows: First, we recall basic definitions and properties of path-based norms (Section 2). In Section 3 we clarify the separable n-dimensional Voronoi map extraction algorithm and for the sake of consistency, we detail the 2D algorithm first proposed in [10]. In Sections 5 and 7, we present and analyse the proposed $n-$dimensional algorithm.

2 Preliminaries

Definition 1 (Norm and metric induced by a norm) Given a vector space EV, a norm is a map g from EV to a sub-group F of $\mathbb{R}$ such that $\forall \mathbf{x},\mathbf{y} \in EV$,

$$\text{(non-negative)} \quad g(\mathbf{x}) \geq 0 \tag{1}$$
$$\text{(identity of indiscernibles)} \quad g(\mathbf{x}) = 0 \Leftrightarrow \mathbf{x} = \mathbf{0} \tag{2}$$
$$\text{(triangular inequality)} \quad g(\mathbf{x}+\mathbf{y}) \leq g(\mathbf{x}) + g(\mathbf{y}) \tag{3}$$
$$\text{(homogeneity)} \quad \forall \lambda \in \mathbb{R}, \quad g(\lambda \cdot \mathbf{x}) = |\lambda| \cdot g(\mathbf{x}) \tag{4}$$

$d(a,b) := g(b-a)$ is the metric induced by the norm g. The triplet (E,F,d), where E is a field, and F a sub-group of $\mathbb{R}$ is called a metric space if $d: E \times E \to F$ (with E such that for $a,b \in E$, $(b-a) \in EV$).

Note that the above definition can be extended from vector spaces to *modules* on a commutative ring ($\mathbb{Z}^n$ being a module on $\mathbb{Z}$ but not a vector space) [29]. Path-based approaches (chamfer masks, -weighted- neighborhood sequences...) aim at defining *digital* metrics induced by norms in metric spaces $(\mathbb{Z}^n, \mathbb{Z}, d)$. Note that (weighted, with $w_i \geq 0$) L_p metrics

$$d_{L_p}(a,b) = \left(\sum_{k=1}^{n} w_k |a_k - b_k|^p \right)^{\frac{1}{p}}, \tag{5}$$

define metric spaces $(\mathbb{Z}^n, \mathbb{R}, d_{L_p})$ which are not *digital*. However, rounding up the distance function $(\mathbb{Z}^n, \mathbb{Z}, \lceil d_{L_p} \rceil)$ is a digital metric space [16].

Definition 2 (Distance Transformation and Voronoi Map) The distance transform DT_X associated with a digital metric space $(\mathbb{Z}^n, \mathbb{Z}, d)$ is a map $X \to \mathbb{Z}$ such that, for $a \in X$ $DT_X(a) = \min_{b \in \mathbb{Z}^n \setminus X}\{d(a,b)\}$. The Voronoi map is the map $X \to \mathbb{Z}^n \setminus X$: $\Pi_X(a) = \arg\min_{b \in \mathbb{Z}^n \setminus X}\{d(a,b)\}$.

The Voronoi map Π_X corresponds to the intersection between the continuous Voronoi diagram for the metric d of points $\mathbb{Z}^n \setminus X$ and the lattice $\mathbb{Z}^n$. If a digital point a belongs to a Voronoi diagram $d-$facet ($0 \leq d < n$), a is equidistant to $n+1-d$ or more points in $\mathbb{Z}^n \setminus X$ but only one is considered in $\Pi_X(a)$ this choice has no influence on DT_X.

Definition 3 (Chamfer Mask) A weighted vector is a pair $(\mathbf{v}, w)$ with $\mathbf{v} \in \mathbb{Z}^n$ and $w \in \mathbb{N}^*$. A chamfer mask $\mathcal{M}$ is a central-symmetric set of weighted vectors with no null vectors and containing at least a basis of $\mathbb{Z}^n$.

In most situations, vectors of a chamfer mask exhibit axial symmetries. We may refer to the *generator* $\mathcal{G}$ *of* $\mathcal{M}$ as the subset of vectors defining $\mathcal{M}$ by symmetries (usually defined in the subspace $x_n \geq \ldots \geq x_1 \geq 0$ of $\mathbb{Z}^n$). Many authors have proposed algorithmic and/or analytic approaches to construct chamfer masks approximating the Euclidean metric. In the following, we focus on *chamfer norms* which are chamfer metrics induced by a norm. To evaluate distances between two digital points for a given chamfer metric, direct formulations have been proposed with simple geometrical interpretation:

Definition 4 (Rational ball, minimal H-representation [29,23]) Given a Chamfer norm $\mathcal{M}$, the rational ball associated with $\mathcal{M}$ is the polytope

$$\mathcal{B}_R = conv\left\{ \frac{\mathbf{v}_k}{w_k};\ (\mathbf{v}_k, w_k) \in \mathcal{M} \right\}. \tag{6}$$

where *conv* denotes the the convex hull. Rational balls for some chamfer masks are illustrated in Figure 1.

The rational ball $\mathcal{B}_R$ can also be described as the H-representation of polytope with minimal parameter [24]: $P = \{x \in \mathbb{Z}^n; Ax \leq y\}$ such that $\forall k \in [1 \ldots f]$, $\exists x \in P \quad A_k x = y_k$.[1] f is the number of rows in A and the number of facets in $\mathcal{B}_R$, and is thus related to $|\mathcal{M}|$.

[1] A_k being the k^{th} row of A.

Table 1 Computational cost summary for separable Voronoi map computation on N^n domains (m being the size of the chamfer norm and f the number of row in a H-representation of the mask, see below).

Metric	CLOSEST	HIDDENBY	Sep. Voronoi Map	Reference
L_2	$O(1)$	$O(1)$	$\Theta(n \cdot N^n)$	[15]
L_∞	$O(1)$	$O(1)$	$\Theta(n \cdot N^n)$	[21]
L_1	$O(1)$	$O(1)$	$\Theta(n \cdot N^n)$	[21]
L_p (exact pred.)	$O(\log p)$	$O(\log p \cdot \log N)$	$O(n \cdot N^n \cdot \log p \cdot \log N)$	Lem. 1
L_p (inexact pred.)	$O(1)$	$O(\log N)$	$O(n \cdot N^n \cdot \log N)$	Lem. 1
2D Chamfer norm	$O(\log m)$	$O(\log^2 m)$	$O(\log^2 m \cdot N^2)$	[10] and Theorem 1
2D Neig. seq. norm	$O(\log m)$	$O(\log^2 m)$	$O(\log^2 m \cdot N^2)$	[24] with Theorem 1
nD Chamfer norm	$O(n + \log f)$	$O((n + log f) \cdot \log N)$	$O(n \cdot N^n \cdot \log N \cdot (n + \log f))$	Corollary 1

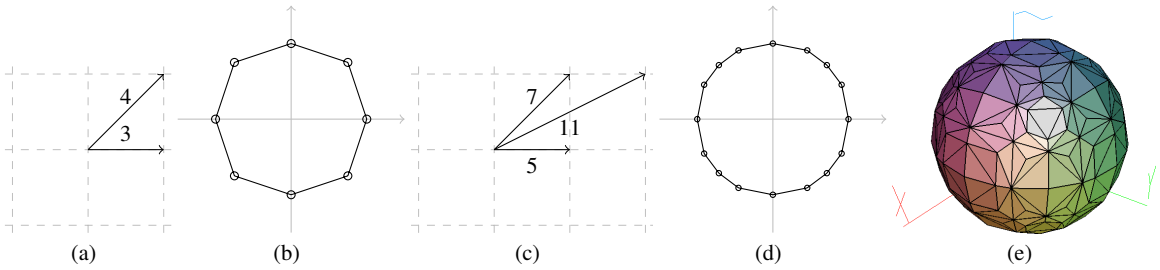

Fig. 1 Chamfer masks and rational balls: in dimension 2, generator vectors for the mask $\mathcal{M}_{3-4}$ (a), its rational ball (b). Generator vectors for $\mathcal{M}_{5-7-11}$ (c) and its rational ball (d). In dimension 3, rational ball of a chamfer mask obtained using generator vectors $(x,y,z) \in [\![-3,3]\!]^3$ and weights computed using [14].

The distance between two points a and b in $\mathbb{Z}^n$ for a chamfer mask is the length shortest path between a and b on a weighted graph $G = (V,E)$ where the vertices are grid points and there is a (weighted) edge between two points if their difference is a vector $\mathbf{v} \in \mathcal{M}$ (the edge is weighted by the associated vector weight) [5]. Since weights are positive integers (see Def. 3), distance values are *scaled* by the weight of the first vector ($(1,0\ldots,0)^T$ by convention). Hence, using masks defined in Fig. 1, $\frac{1}{3} \cdot d_{\mathcal{M}_{3-4}}(a,b)$ and $\frac{1}{5} \cdot d_{\mathcal{M}_{5-7-11}}(a,b)$ are approximations of $d_{L_2}(a,b)$.

An important result for distance computation can be summarized as follows:

Proposition 1 (Direct Distance Computation [23]) *Given a chamfer mask $\mathcal{M}$ induced by a norm and (A,y) its minimal parameter H-representation, then for any $a \in \mathbb{Z}^n$, the chamfer distance of a from the origin is*

$$d_{\mathcal{M}}(O,a) = \max_{1 \le k \le f} \{A_k a^T\}. \tag{7}$$

Among path-based digital metric, (weighted) neighborhood sequences have been proposed to have better approximation of the Euclidean metric from sequences of elementary chamfer masks [26,22,28,24]. A key result has been demonstrated in [24] stating that for such distance functions, a minimal parameter polytope representation exists and that distances can be obtained from a expression similar to (7):

$$d(O,a) = \max_{1 \le k \le f} \{f_k(A_k a^T)\}, \tag{8}$$

f_k being some integer sequence characterizing the neighborhood sequence metric. In the following and for the sake of simplicity, we describe our algorithms focusing on chamfer norms but similar results can be obtained for more generic path-based metrics such as neighborhood sequences.

3 Separable distance transformation

3.1 Voronoi map from separable approach and metric conditions

In [15,7,21,20], several authors have described optimal in time and separable techniques to compute error-free Voronoi maps or distance transformations for L_2 and L_p metrics. Separability means that computations are performed dimension by dimension. In the following, we consider the *Voronoi Map* approach as defined in [7]. Let us first define an hyper-rectangular image $I_X : [1..N_1] \times \ldots \times [1..N_n] \to \{0,1\}$ such that $I_X(a) = 1$ for $a \in [1..N_1] \times \ldots \times [1..N_n]$ iff $a \in X$ ($I_X(a) = 0$ otherwise). The separable algorithm that computes the Voronoi Map for I_x is defined in Algorithm 1. First, the Voronoi map is initialized by processing each span[2] of the input image along the first dimension in order to create independent 1D Voronoi maps for the metric (lines $5-6$). Then, for each further dimension, the partial Voronoi map

[2] An image span S along the q^{th} direction is a vector of N_q points with same coordinates except at their q^{th} one.

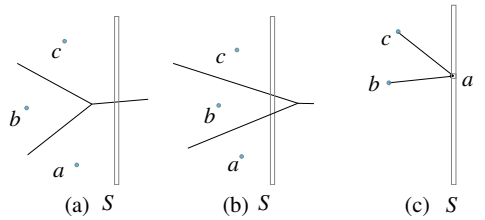

Fig. 2 Geometrical predicates for Voronoi map construction: HIDDENBY(a,b,c,S) returns true in (a) and false in (b) (straight segments correspond to Voronoi diagram edges). (c) illustrates the CLOSEST(a,b,c) predicate for $c \in S$.

Π_X is updated using one dimensional independent processes on each span along the q^{th} dimension (line 8). Algorithm 2 describes the function VORONOIMAPSPAN. This function is the core of the separable algorithm as it defines the 1D processes to perform on each row, column and higher dimensional image span. In this process, metric information are embedded in the following key predicates (see Fig. 2):

1. CLOSEST(a,b,c): given three points $a,b,c \in \mathbb{Z}^n$ this predicate returns true if $d(a,b) < d(a,c)$;
2. HIDDENBY(a,b,c,S): given three points $a,b,c \in \mathbb{Z}^n$ such that $a_q < b_q < c_q$[3] and a 1D image span S, this predicates returns true if there is no $s \in S$ such that

$$d(b,s) < d(a,s) \text{ and } d(b,s) < d(c,s)\,. \tag{9}$$

Algorithm 1: VORONOIMAP(BINARY MAP I_X)

```
1 Π_X = empty image, same size as I_X;
2 for q in {1...n} do
3     for (x_1,..x_{q-1},x_{q+1},..x_n) in
      [1..N_1] × ..[1..N_{q-1}] × [1..N_{q+1}].. × [1..N_n] do
4         S = {s^i}_{i∈[1..N_q]} where s^i = (x_1..x_{q-1},i,x_{q+1}..x_n);
          // all the coordinates are fixed in S
             except the q^th one
5         if q == 1 then
              // Π_X is initialized span by span
6             Π_X = Π_X ∪ VORONOIMAPSPAN(I_X, q, S);
7         else
              // Π_X is updated along span S
8             Π_X = VORONOIMAPSPAN(Π_X, q, S);
9 return Π_X
```

In other words, HIDDENBY returns true if and only if the Voronoi cells of sites a and c *hide* the Voronoi cell of b along S. For L_1, L_2 and L_∞ metrics, CLOSEST and HIDDENBY predicates can be computed in $O(1)$ [7,15,21]. Hence, Algorithm 2 is in $O(N_q)$ for dimension q, leading to an overall computational time for the Voronoi Map (Algorihtm 1) and

[3] Subscript a_q denotes the q^{th} coordinate of point $a \in \mathbb{Z}^n$.

Algorithm 2: VORONOIMAPSPAN(MAP $\mathcal{M}_X$, DIMENSION q, 1D SPAN S)

```
Data: q is an integer in {1...n};
      S is a 1D span along dimension q, with points {s^1,...,s^{N_q}}
      sorted by their q^th coordinate;
      M_X is either a binary map if q = 1 or a partial Voronoi Map.
Result: Partial Voronoi map Π_X updated along S.
1  if q == 1 ; // Special case for the first dimension
2  then
3      Π_X = empty image, same size as M_X;
4      k = 0;
5      foreach point s in S do
6          if M_X(s) == 0 then
7              L_S[k] = s;
               // L_S = list of the sites visible on S
8              k++;
9  else
10     Π_X = M_X;
11     L_S[0] = M_X(s^1);
12     L_S[1] = M_X(s^2);
13     k = 2 , l = 3;
       // Update the list L_S
14     while l ≤ N_q do
15         w = M_X(s^l);
16         while k ≥ 2 and HIDDENBY(L_S[k-1], L_S[k], w, S) do
               // L_S[k] is no longer visible, unstack
17             k--;
18         k++ ; l++;
19         L_S[k] = w;
20 foreach point s in S by increasing q^th coordinate do
21     while (k < |L_S|) and CLOSEST(s, L_S[k+1], L_S[k]) do
           // s is closer to L_S[k+1], look further
22         k++ ;
23     Π_X[s] = L_S[k];
24 return Π_X
```

Distance Transformation computations in $\Theta(n \cdot N^n)$ (if we assume that $\forall q \in [1\ldots n], N_q = N$).

In [15] or [20], authors discussed about conditions on the metric d to ensure that Algorithm 2 is correct. The key property can be informally described as follows: given two points $a,b \in \mathbb{Z}^n$ such that $a_q < b_q$ and a straight line l along the q^{th} direction and if we denote by $v_l(a)$ (resp. $v_l(b)$) the intersection between the Voronoi cell of a (resp. b) and l, then $v_l(a)$ and $v_l(b)$ are simply connected Euclidean segments and $v_l(a)$ appears *before* $v_l(b)$ on l (so called *monotonicity property* in [20] and is related to *quadrangle inequality* in [15]). These contributions are summed up the Definition 5 and Proposition 2.

Definition 5 (Axis symmetric ball norm) A metric d induced by a norm whose unit ball is symmetric with respect to grid axes is called *axis symmetric ball norm*.

Proposition 2 (Metric conditions [15]) *Algorithm 1 exactly computes the Voronoi Map Π_X of a binary input image I_X for any axis symmetric ball norm.*

Proposition 2 implies that most chamfer norms and neighborhood sequence based norms can also be considered in separable Algorithm 1 (see Fig. 3). However, note that Algorithm 2, and as a by-product Algorithm 1, are exact only if the distance comparison predicate is exact, i.e. if we can compare two distances, through the CLOSEST and HIDDENBY predicates, without error.

Furthermore, computational efficiency of the algorithm requires the design of efficient algorithmic tools to implement these predicates, and this the purpose of the next sections.

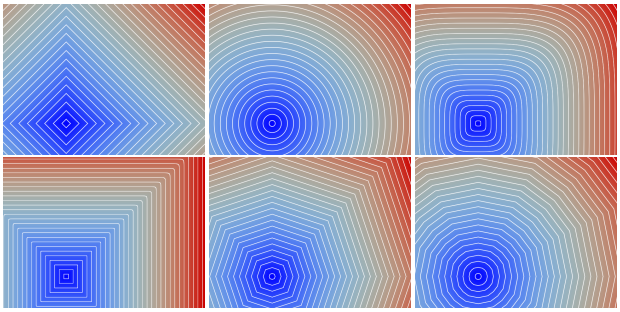

Fig. 3 Distance transformation from a single source for different metrics satisfying Definition 5 and thus Proposition 2: *(from left to right)* L_1, L_2, L_4, L_{80}, $\mathcal{M}_{3-4}$ and $\mathcal{M}_{5-7-11}$.

3.2 Generic predicates and Complexity analysis for axis symmetric ball norms

We first detail the overall computational cost of Algorithms 2 and 1. We assume in the following that $\forall q \in [1 \dots n], N_q = N$.

Lemma 1 **([10])** *Let $(\mathbb{Z}^n, F, d)$ be a metric space induced by a norm with axis symmetric unit ball. If C denotes the computational cost of* CLOSEST *predicate and H is the computational cost of the* HIDDENBY *predicate, then Algorithm 2 is in $O(N \cdot (C+H))$, leading to a complexity of $O(n \cdot N^n \cdot (C+H))$ for Algorithm 1.*

For a given axis symmetric ball norm d, we first define generic Algorithms 3, 4 and 5. Note that these algorithms are valid for any dimension n. The computational cost of the CLOSEST predicate is simply the one of a distance evaluation. As a first approach, Algorithms 4 and 5 show that the HIDDENBY predicate can be obtained by a binary search on the 1D image span S to localize the abscissa of Voronoi edges of sites $\{a,b\}$ and $\{b,c\}$ (see Fig. 4).

The complexity H of Algorithm 5 can be expressed as a function of the complexity C of Algorithm 3, leading to the general result below:

Algorithm 3: Generic CLOSESTND($a,b,c \in \mathbb{Z}^n$).

```
return d(a,b) < d(a,c);
```

Algorithm 4: Generic VORONOIEDGE($a,b,s^i,s^j \in \mathbb{Z}^n$) with $i < j$, $a_q < b_q$.

```
if (j − i = 1) then
    if i = 1 and CLOSEST(s^i, b, a) then
        return −∞;
    if i = N_i and CLOSEST(s^i, a, b) then
        return ∞;
    return i;
mid = i + (j − i)/2;
if CLOSEST(s^mid, a, b) then
    // s_mid closer to a
    return VORONOIEDGE(a, b, s^mid, s^j)
else
    // s_mid closer to b
    return VORONOIEDGE(a, b, s^i, s^mid)
```

Algorithm 5: Generic HIDDENBY($a,b,c \in \mathbb{Z}^n; S$ in the q^{th} direction) with $a_q < b_q < c_q$.

```
v_ab = VORONOIEDGE (a, b, s^1, s^{N_q});
v_bc = VORONOIEDGE (b, c, s^1, s^{N_q});
return (v_ab > v_bc);
```

Lemma 2 **([10])** *Let $\mathcal{M}$ be a chamfer norm with axis symmetric unit ball in dimension n whose rational ball has f facets, Algorithm 1 can be implemented with a computational complexity of $O(n \cdot N^n \cdot C \cdot \log N)$, where N^n is the size of the image.*

Although similar approaches could be contemplated for other path-based metrics, in the following we focus on chamfer norms. Section 3 focuses on dimension 2 while Section 5 tackles the problem in higher dimensions.

4 Sublinear algorithm in dimension 2 for chamfer norms

Let us consider a 2D chamfer norm $\mathcal{M}$ with m weighted vectors (note that $f := |\mathcal{B}_R| = m$ in 2D). We suppose that vectors $\{\mathbf{v}^k\}_{k=1\dots m}$ are sorted counterclockwise. We define a wedge as a pair $(\mathbf{v}^k, \mathbf{v}^{k+1})$ of vectors. To each wedge is associated a row A_k in the minimal H-representation of A (A_k can also be seen as a –non-unitary– normal vector to $\mathcal{B}_R$ facets [23]). Using similar notations, [29,28] demonstrate that the distance evaluation of point a can be obtained in two steps: first, we compute the wedge $(\mathbf{v}^k, \mathbf{v}^{k+1})$ a belongs to. Then,

$$d_{\mathcal{M}}(O,a) = A_k \cdot a^T \,. \tag{10}$$

Lemma 3 ([10]) *Given a chamfer norm $\mathcal{M}$ in dimension 2 with m vectors, the distance computation and thus the* CLOSEST *predicate are in $O(\log m)$.*

Lemma 4 extends this results to higher dimensions. To optimize the HIDDENBY predicate, we need to focus on the VORONOIEDGE function. Given two points a and b ($a_q < b_q$) and a 1D image span S along the q^{th} dimension, we have to find the abscissa $e_q \in \mathbb{Z}$ of the point $e \in \mathbb{Z}^n$ on S such that all the points of S of abscissa lower than e_q are in the Voronoi cell of a while all the points with a greater abscissa are in the Voronoi cell of b. Let us first suppose that we do not know e but we know the wedge $(\mathbf{v}^k, \mathbf{v}^{k+1})$ (resp. $(\mathbf{v}^j, \mathbf{v}^{j+1})$) the vector $(e-a)^T$ (resp. $(e-b)^T$) belongs to (see Fig. 4−(c)). In this situation, we know that e is the solution of

$$A_k \cdot (e-a)^T = A_j \cdot (e-b)^T \ , \tag{11}$$

(since $e \in S$, we have one linear equation with only one unknown e_i). As a consequence, if we know the two wedges the Voronoi edge belongs to, we have the abscissa in $O(1)$ (see Algorithm 6 and Fig. 4−(c)).

Algorithm 6: 2D chamfer norm VORONOIEDGE($a, b \in \mathbb{Z}^2$, span S, chamfer norm $\mathcal{M} = \{\mathbf{v}^k\}_{k=1\dots m}$).

1 $(\mathbf{v}^k, \mathbf{v}^{k+1})$ = VORONOIEDGEWEDGE$(a, b, \mathbf{v}^1, \mathbf{v}^m, S)$;
2 $(\mathbf{v}^j, \mathbf{v}^{j+1})$ = VORONOIEDGEWEDGE$(b, a, \mathbf{v}^1, \mathbf{v}^m, S)$;
3 Compute the abscissa e_q of the point e such that $A_k \cdot (e-a)^T = A_j \cdot (e-b)^T$;
4 **return** e_q;

To obtain both wedges, we use a binary search similar to Algorithm 4: Algorithm 7 returns the wedge associated with a containing the Voronoi edge with respect to b. Applying this algorithm to obtain the wedge associated with b with respect to a defines Algorithm 6. The binary search shrinks the set of vectors $\{\mathbf{v}^i, \dots, \mathbf{v}^j\}$ to end up with a wedge $(\mathbf{v}^k, \mathbf{v}^{k+1})$ such that the intersection point between the straight line $(a+\mathbf{v}^k)$ and S is in the Voronoi cell of b and such that the intersection between $(a+\mathbf{v}^{k+1})$ and S is in the Voronoi cell of a (see Fig. 4−(c)). Algorithm 7 thus first computes the intersection points associated with a wedge $(\mathbf{v}^{i+(j-i)/2}, \mathbf{v}^{i+(j-i)/2+1})$ (lines 5 – 6); evaluates the distances at these points (lines 7 – 10) and then decides which set $\{\mathbf{v}^i, \dots, \mathbf{v}^{i+(j-i)/2}\}$ or $\{\mathbf{v}^{i+(j-i)/2}, \dots, \mathbf{v}^j\}$ has to be considered for the recursive call (lines 14 – 20 and Fig. 4−(b)).

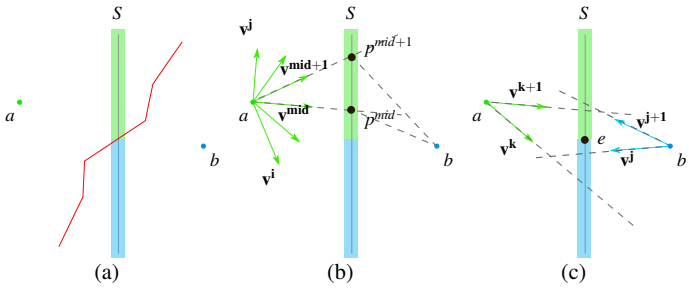

Fig. 4 VORONOIEDGEWEDGE and VORONOIEDGE: (a) initial problem, we want to compute the intersection between S and the Voronoi edge of a and b (in red). (b) an internal step of VORONOIEDGEWEDGE to reduce the set of directions of $\mathcal{M}$ at a (here the next recursive call will be on $(\mathbf{v}^i, \mathbf{v}^{mid})$).$(c)$ final step of VORONOIEDGE where both wedges have been obtained and thus e can be computed.

Algorithm 7: VORONOIEDGEWEDGE($a, b \in \mathbb{Z}^2; \mathbf{v}^i, \mathbf{v}^j$ in $\mathcal{M}$; span S in dimension q), with $i < j$.

1 **if** $(j-i=1)$ **then**
2 **return** $(\mathbf{v}^i, \mathbf{v}^{i+1})$;
3 **else**
4 $mid = i + (j-i)/2$;
5 Let p^{mid} be the intersection point between $(a+\mathbf{v}^{mid})$ and S;
6 Let p^{mid+1} be the intersection point between $(a+\mathbf{v}^{mid+1})$ and S;
  // $O(1)$ evaluation of distances w.r.t. a
7 $d^a_{p^{mid}} = A_{mid} \cdot (p^{mid} - a)^T$;
8 $d^a_{p^{mid+1}} = A_{mid+1} \cdot (p^{mid+1} - a)^T$;
  // $O(\log m)$ evaluation of distances w.r.t. b
9 $d^b_{p^{mid}} = d_{\mathcal{M}}(b, p^{mid})$;
10 $d^b_{p^{mid+1}} = d_{\mathcal{M}}(b, p^{mid+1})$;
11 Let b_{mid} be true if $d^a_{p^{mid}} < d^b_{p^{mid}}$; false otherwise;
12 Let b_{mid+1} be true if $d^a_{p^{mid+1}} < d^b_{p^{mid+1}}$; false otherwise;
13 **if** $b_{mid} \neq b_{mid+1}$; // we found the Voronoi edge wedge
14 **then**
15 **return** $(\mathbf{v}^{mid}, \mathbf{v}^{mid+1})$;
16 **if** $b_{mid} = b_{mid+1} = true$; // Both points are in a's cell
17 **then**
18 **if** $a_q < b_q$ **then**
19 **return** VORONOIEDGEWEDGE$(a, b, \mathbf{v}^{mid}, \mathbf{v}^j, S)$;
20 **else**
21 **return** VORONOIEDGEWEDGE$(a, b, \mathbf{v}^i, \mathbf{v}^{mid}, S)$;
22 **if** $b_{mid} = b_{mid+1} = false$; // Both points are in b's cell
23 **then**
24 **if** $a_q < b_q$ **then**
25 **return** VORONOIEDGEWEDGE$(a, b, \mathbf{v}^i, \mathbf{v}^{mid}, S)$;
26 **else**
27 **return** VORONOIEDGEWEDGE$(a, b, \mathbf{v}^{mid}, \mathbf{v}^j, S)$;

Theorem 1 ([10]) *Let $\mathcal{M}$ be a 2D chamfer norm with axis symmetric unit ball and m weighted vectors, then we have: (i) Algorithm 6 is in $O(\log^2 m)$; (ii) Algorithm 2 (with predicates from Algorithm 6 and Lemma 3), computes a Voronoi map Π_X and thus the distance transformation of X for metric $d_{\mathcal{M}}$ in $O(\log^2 m \cdot N^2)$.*

In the next section, we extend all these results to higher dimension. First, we use the combinatorics of the rational ball to design an efficient CLOSEST predicate. Then, we show that the 2D VORONOIEDGE principle naturally arises in the *nD* case.

5 Distance transformation algorithm in higher dimension for chamfer norms

5.1 CLOSEST predicate and first results

Let us consider a general chamfer norm in arbitrary dimension n. First, let us discuss about chamfer mask combinatorics. If m denotes the number of weighted vectors of $\mathcal{M}$, its rational ball $\mathcal{B}_R$ has $O(m^{\lfloor \frac{n}{2} \rfloor})$ $i-$facets $(0 \leq i \leq d)$ [13]. If we denote by f the number of $(n-1)-$facets of $\mathcal{B}_R$ (*i.e.* number of row in the H-representation of $\mathcal{B}_R$), we have by duality the result that $|\mathcal{B}_R| = O(f^{\lfloor \frac{n}{2} \rfloor})$. As a consequence, we have the following distance evaluation result:

Lemma 4 *Let $\mathcal{M}$ be a chamfer norm whose rational ball $\mathcal{B}_R$ has f $(n-1)-$facets in dimension n, then distance computation and thus* CLOSEST *predicate are in (amortized) $O(n+\log f)$ with $O\left(\frac{f^{\lfloor \frac{n}{2} \rfloor}}{(\log f)^{\lfloor \frac{n}{2} \rfloor - \delta}}\right)$ space and preprocessing time*[4].

Proof Similarly to the 2D case, the distance $d_{\mathcal{M}}(O,a)$ for $a \in \mathbb{Z}^n$ is given by first solving a ray-shooting problem on convex polytopes which consists in first computing the $(n-1)$-facet of $\mathcal{B}_R$ pierced by the ray (O,a). Once the facet is obtained, the associated A_k row is used to evaluate $d_{\mathcal{M}}(O,a) = A_k \cdot a^T$ in $O(n)$. Following [19], such a ray-shooting query on convex polytopes can be solved in $O(\log f)$ thanks to a preprocessing in $O\left(\frac{f^{\lfloor \frac{n}{2} \rfloor}}{(\log f)^{\lfloor \frac{n}{2} \rfloor - \delta}}\right)$. In the case when the ray hits a facet of dimension strictly lower than $n-1$, the algorithm returns one of the adjacents $(n-1)$-facets. Propositions 3 and 4 from [23] ensure that the choice of any $(n-1)$-facet leads to the same distance evaluation. Please note also that the preprocessing time is roughly equivalent to the convex hull computation in higher dimension which is in $O(f^{\lfloor \frac{n}{2} \rfloor})$. Hence, preprocessing for ray-shooting can be done while computing the rational ball $\mathcal{B}_R$ using Eq. (6).□

Algorithm 4 being valid in any dimension, we can use Corollary 2 to straightforwardly obtain the result below:

Corollary 1 *Let $\mathcal{M}$ be a chamfer norm whose rational ball $\mathcal{B}_R$ has f $(n-1)-$facets in dimension n, separable exact Voronoi Map Π_X can be obtained in $O(n \cdot N^n \cdot \log N \cdot (n + \log f))$, thanks to a preprocessing in $O\left(\frac{f^{\lfloor \frac{n}{2} \rfloor}}{(\log f)^{\lfloor \frac{n}{2} \rfloor - \delta}}\right)$.*

[4] δ is an arbitrarily small positive constant.

However, we show below that we can still expect faster VORONOIEDGE function even in higher dimension.

5.2 Improved HIDDENBY predicate algorithm

In dimension 2, we have shown that Algorithm 7 enables to lower down the complexity from a logarithmic factor on the size N of the image to a logarithmic factor on the size m of the mask. In this algorithm, binary search is performed on a set of vectors going from a point a to points (denoted by p^{mid} and p^{mid+1}) on a span S. Whatever the dimension n of the image, these vectors lie in the smallest affine subspace containing S, which is a one-dimensional flat, and the point $a \in \mathbb{Z}^n$. In the general case where the point does not lie on the span, this is actually always a 2-flat (see Fig. 5).

Computing the intersection of this 2-flat with the rational ball $\mathcal{B}_R$ comes down to intersecting a n-polytope with $n-2$ hyperplanes. By duality, each of these operations is equivalent to a convex hull computation, with a complexity of $\mathcal{O}(f^{\lfloor n/2 \rfloor})$ [8,2]. As a consequence, in order to make the approach efficient, we must avoid to compute explicitely this intersection.

To do so, in Algorithm 8 we rewrite the VORONOIEDGEWEDGE function, as a modification of Algorithm 7. Recall that the goal of this function is to find the $n-1$-facet (wedge) of the rational ball containing the vector $\mathbf{v}$ such that $(a+\mathbf{v}) \cap S$ is on the bisector of a and b. Let us first give some notations. Let $\mathcal{P}$ be the 2-flat defined by S and a. We denote by $\mathcal{F}^a_{\mathbf{v}}$ the result of the ray shooting of vector $a+\mathbf{v}$ on the rational ball $\mathcal{B}_R$ centered in a. $A_{\mathcal{F}}$ denotes the row of matrix A corresponding to the $n-1$ facet $\mathcal{F}^a_{\mathbf{v}}$.

The algorithm computes two vectors v_i and v_j of $\mathcal{P}$ such that $e_i = (a+v_i) \cap S$ belongs to the Voronoi cell of a, $e_j = (a+v_j) \cap S$ to the Voronoi cell of b and $\mathcal{F}^a_{\mathbf{v}_i} = \mathcal{F}^a_{\mathbf{v}_j}$. Note that, contrary to Algorithm 7 vectors v_i, v_j are not necessarily chamfer vectors anymore, but by construction, they always lie in $\mathcal{P}$. Finally, the boolean a-IS-LOW is true when s^1 is in a's Voronoi cell, false otherwise.

Two invariants are maintained in Algorithm 8 : (i) e_i is lower than e_j on span S ($e_{i_q} < e_{j_q}$) ; (ii) if a-IS-LOW is true, e_i in a's Voronoi cell, e_j in b's Voronoi cell, and conversely if a-IS-LOW is false.

The first test on line 1 ends the recursive call when the wedge of the bisector has been found. If the test on line 5 is true, the recursion ends: indeed, in this case, points $\lfloor e_{i_q} \rfloor$ and $\lceil e_{j_q} \rceil$ are successive points of S, and from invariants (i) and (ii), we know that $\lfloor e_{i_q} \rfloor$ lies in the same Voronoi cell as e_i, and $\lceil e_{j_q} \rceil$ in the same Voronoi cell as e_j, which are different by invariant (ii) and precondition of Algorithm 9.In this case, the intersection between the bisector and S is actually equal to $\lfloor e_{i_q} \rfloor$. Lines $8-23$ of the algorithm is the core of

the binary search process: vector $\mathbf{v}$ is defined as the bisector of $\mathbf{v}_i$ and $\mathbf{v}_j$ and we test whether the intersection between the line $(a+\mathbf{v})$ and S lies in the Voronoi cell of a or b.

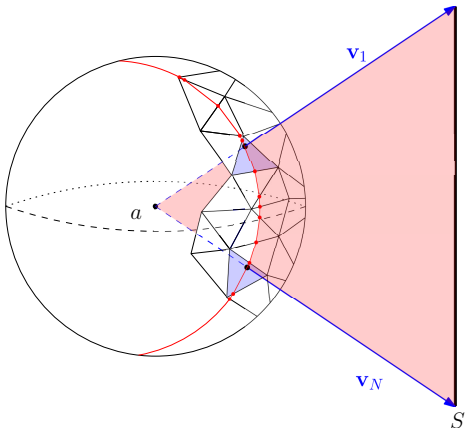

Fig. 5 Vectors $\mathbf{v}_i$ lie in a 2-flat defined by S and a, in light red. The distance $d_{\mathcal{M}}$ between a point on S and a is computed via a ray shooting that returns the $n-1$-facet of $\mathcal{B}_R$ traversed by the ray : the facets $\mathcal{F}^a_{\mathbf{v}_1}$ and $\mathcal{F}^a_{\mathbf{v}_N}$ are depicted in light blue.

The nD variant of the VORONOIEDGE function (Algorithm 6) is then pretty straightforward : first, lines 1 to 9 are dedicated to the initialization of the boolean a-IS-LOW ; then it is enough to call the VORONOIEDGEWEDGEND function and replace the wedge $(\mathbf{v}^k, \mathbf{v}^{k+1})$ by its nD counterpart $\mathcal{F}_k$.

Corollary 2 *Let $\mathcal{M}$ be a chamfer norm in dimension n whose rational ball $\mathcal{B}_R$ has f $(n-1)-$facets. Let W be the computational time complexity of the* VORONOIEDGEWEDGEND *function. Then, the separable exact Voronoi Map can be obtained in (amortized) $O(n\cdot N^n\cdot(n+\log f+W))$ with a $O\left(\frac{f^{\lfloor\frac{n}{2}\rfloor}}{(\log f)^{\lfloor\frac{n}{2}\rfloor-\delta}}\right)$ space and preprocessing time. More precisely, the worst-case complexity W being $\mathcal{O}((n+\log f)\cdot\log N)$, this leads to a global (amortized) complexity of $O(n\cdot N^n\cdot\log N\cdot(n+\log f))$ (same preprocessing).*

Proof Following Lemma 1, the generic separable algorithms computes the Voronoi map in $O(n\cdot N^n\cdot(C+H)$. Corollary 4 states that $C=O(n+\log f)$ with a $O\left(\frac{f^{\lfloor\frac{n}{2}\rfloor}}{(\log f)^{\lfloor\frac{n}{2}\rfloor-\delta}}\right)$ space and preprocessing time. Remains to evaluate H, *i.e.* the complexity of the VORONOIEDGEND function. In Algorithm 9, the first eight lines and the verification of the precondition are in $O(C)$ since only distance computations are involved. Lines 10 and 12 are calls to the VORONOIEDGEWEDGEND function, with a complexity in $O(W)$. In the worst case, we have $W=\mathcal{O}((n+\log f)\cdot\log N)$ thanks to the test on line 5. Last, the system to solve in line 13 has only one unkwown e_q since e belongs to the one-dimensional span S, with a complexity of $O(1)$. □

Algorithm 8: VORONOIEDGEWEDGEND($a,b\in\mathbb{Z}^n;\mathbf{v}_i,\mathbf{v}_j$ in $\mathcal{P}$; S along the q^{th} direction; $\mathcal{F}_{\mathbf{v}_i}$, $\mathcal{F}_{\mathbf{v}_j}$; boolean a-IS-LOW)

```
1  if F_{v_i} = F_{v_j} then
2      return F_{v_i};
3  else
4      e_i = (a+v_i) ∩ S, e_j = (a+v_j) ∩ S;
5      if ⌈e_{j_q}⌉ = ⌊e_{iq}⌋ + 1 then
6          return F_{v_i};
7      else
8          v = v_i/‖v_i‖ + v_j/‖v_j‖;
9          p = (a+v) ∩ S;
               // O(n+log f) evaluation of distances
           w.r.t. a and b
10         d^a_p = A_{F^a_v} · (p−a)^T;
11         d^b_p = A_{F^b_{(p−b)}} · (p−b)^T;
12         Let IN-a be true if d^a_p < d^b_p; false otherwise;
13         if IN-a = true then
14             if a-IS-LOW then
15                 return
                   VORONOIEDGEWEDGE(a,b,v,v_j,S,F^a_v,F_{v_j})
16             else
17                 return
                   VORONOIEDGEWEDGE(a,b,v_i,v,S,F_{v_i},F^a_v)
18         else
19             if a-IS-LOW then
20                 return
                   VORONOIEDGEWEDGE(a,b,v_i,v,S,F_{v_i},F^a_v)
21             else
22                 return
                   VORONOIEDGEWEDGE(a,b,v,v_j,S,F^a_v,F_{v_j})
```

Note that in the worst-case, this approach does not improve the result presented in Corollary 1 (using the generic VORONOIEDGE of Algorithm 4). However, in Section 7, we give some experimental insights on a finer analysis of the complexity W under distribution hypothesis.

6 L_p metric case

As a direct consequence of Lemma 1, we briefly derive computational costs for L_p metrics. For such metrics, the CLOSEST and HIDDENBY predicates are in $O(1)$ for $p=\{1,2,\infty\}$ with exact integer only computations [20,21]. We thus have distance transformation algorithms in $\Theta(n\cdot N^n)$.

For $p\in\mathbb{R}$, $p\geq 1$, we can use approximations of the evaluation of distances on IEEE 754 *double* and then consider the Generic HIDDENBY predicate in $O(\log N)$ (Alg. 5). As predicates being based on floating point computations, numerical issues may occur but we have an $O(n\cdot N^n\cdot\log N)$ distance transformation algorithm (L_p inexact predicates in Table 1).

If $p\in\mathbb{Z}$, $p\geq 3$, we use exact integer number based computations of distances storing sum of power p quantities

Algorithm 9: VORONOIEDGEND($a, b \in \mathbb{Z}^n$, span S).

Precondition: the bisector of a and b intersects the span S.

if $a_q < b_q$ **then**
  a-IS-LOW = true
else
  if $b_q < a_q$ **then**
    a-IS-LOW = false
  else
    ; // Compute Voronoi cells of s^1 and s^N.
    $d^a_{s^1} = A_{\mathcal{F}^a_{(s^1-a)}} \cdot (s^1-a)^T$; $d^b_{s^1} = A_{\mathcal{F}^b_{(s^1-b)}} \cdot (s^1-b)^T$;
    a-IS-LOW $= (d^a_{s^1} \leq d^b_{s^1})$; // s^1 is in a's Voronoi cell
    ;
$\mathbf{v_1} = s^1 - a, \mathbf{v_N} = s^N - a$;
$\mathcal{F}_k =$ VORONOIEDGEWEDGEND$(a, b, \mathbf{v}_1, \mathbf{v}_N, \mathcal{F}_{\mathbf{v}_1}, \mathcal{F}_{\mathbf{v}_N}, S,$a-IS-LOW$)$;
$\mathbf{v_1} = s^1 - b, \mathbf{v_N} = s^N - b$;
$\mathcal{F}_j =$ VORONOIEDGEWEDGEND$(b, a, \mathbf{v}_1, \mathbf{v}_N, \mathcal{F}_{\mathbf{v}_1}, \mathcal{F}_{\mathbf{v}_N}, S, \neg($a-IS-LOW$))$;

Compute the abscissa e_q of the point $e \in S$ such that $A_k \cdot (e-a)^T = A_j \cdot (e-b)^T$;
return e_q;

(which can be computed in $O(\log p)$ thanks to exponentiation by squaring). The HIDDENBY predicate is also based on Algorithm 5, leading to an $O(n \cdot N^n \cdot \log p \cdot \log N)$ distance transformation algorithm (L_p exact predicates in Table 1).

7 Experimental analysis

7.1 Insights on the complexity in dimension n

The complexity W of Algorithm 9 depends on the number of recursion steps done until the two vectors stab the same $(n-1)$-face of the rational ball $\mathcal{B}_R$. Thus, this complexity depends on the distribution of the chamfer vectors defining $\mathcal{B}_R$.

Let us denote by P the intersection between the $(n-1)$-faces of $\mathcal{B}_R$ and the 2-flat $\mathcal{P}$ (see the red polygon on Fig. 5). Note that $\mathcal{P}$ goes through the center of $\mathcal{B}_R$. If we assume that the vectors defining $\mathcal{B}_R$ are uniformely distributed on the unit sphere $\mathbb{S}^n$ and that the faces of P are also uniformely distributed on $\mathcal{B}_R \cap \mathcal{P}$, then we can expect that $W = \mathcal{O}((n+\log f) \cdot \log |P|)$. Even if studying precisely these questions is out of scope of this work, in the following we give insights on both the relevance of these assumptions and the behaviour of $|P|$ in the context of chamfer norms.

7.1.1 Some observations on the distribution hypothesis

Following [29,14], a classical way of defining chamfer norms is to consider a set of vectors defined from a subset of Farey sequences. Recall that the Farey sequence $\mathscr{F}^n_m$ of dimension n and order m is defined as follows : $\mathscr{F}^n_m = \{(\frac{x_2}{x_1}, \ldots, \frac{x_n}{x_1}), \gcd_{i \in 1..n}(x_i) = 1, 0 \leq x_n \leq x_{n-1} \leq \cdots \leq x_1 \leq m\}$. Then a Farey sequence $\mathscr{F}^n_m$ is in bijection with all the points $(x_1, \ldots, x_n)$ in $\mathbb{Z}^n$, $0 \leq x_n \leq \cdots \leq x_1 \leq m$ visible from the origin[5]. The vectors $\mathbf{v}_k$ of a chamfer norm in dimension n can be defined using a subset of a particular $\mathscr{F}^n_m$: the weights w_k are set so that the rational ball $\mathcal{B}_R$ is convex. By construction, such chamfer masks are norms with axis symmetric unit balls.

Studying the distribution of such sets of vectors is a field of research in itself, and we simply mention below several results relevant to our context.

First, it is well-known that [18,17] n-dimensional lattice points visible from the origin have a constant density in $\mathbb{R}^n$. Moreover, in [4] the authors study in the 2D case the distribution of the angles of straight lines from the origin through visible points. More precisely, they study the proportion of differences between consecutive angles which are larger than the average: they show that this proportion is smaller than what is expected for a random distribution, and give an explicit formulation of the repartition function. Similar results in higher dimension remain an open question.

These results tend to support the hypothesis of a uniform distribution of the vectors of $\mathcal{B}_R$, but the question of the distribution of the faces of the polygon P has not been investigated to our knowledge.

7.1.2 Experimental behaviour of $|P|$

In this part, we investigate the number of faces of P when $\mathcal{B}_R$ is a rational ball defined from Farey Sequences. The results are presented in Figure 6 and we detail below how the rational balls are generated, how the 2-flats $\mathcal{P}$ are selected, and how the intersection between $\mathcal{B}_R$ and $\mathcal{P}$ is performed.

In the four subfigures of Figure 6, rational balls are defined from Farey sequences:

- In (b-d), the vectors of $\mathcal{B}_R$ are all normalized vectors of a Farey sequence of order m (the higher the order, the greater the number of vertices - and $(n-1)$-faces - of $\mathcal{B}_R$). The order of the Farey sequences ranges from 1 to 10 in (b-c), from 1 to 6 in (d);
- in (a), $\mathcal{B}_R$ is computed thanks to the algorithm presented in [14].[6] Given a (odd) mask size m, and a maximal error ε, the algorithm computes a subset of vectors of $\mathcal{F}_{\frac{m-1}{2}}$ and weights such that the rational ball $\mathcal{B}_R$ is convex and the error with respect to the optimal theoretical error expected (wrt the Euclidean distance) for this mask size is below ε.

[5] A point $p \in \mathbb{Z}^n$ is visible from the origin in $\mathbb{Z}^n$ if there is no point of $\mathbb{Z}^n$ on (Op) between O and p.

[6] Code is available on the TC18 website www.tc18.org/code_data_set/code.php

Once the sets of vectors defined, we use Qhull [3] to compute both the rational ball itself and its intersection with a 2-flat $\mathcal{P}$ that goes through the center of $\mathcal{B}_R$. This intersection is performed by randomly picking the coefficients of $n-2$ $(n-1)$-hyperplanes containing the center of $\mathcal{B}_R$, and iteratively adding each $(n-1)$-hyperplane. The vertices of P are the points lying on all $(n-1)$-hyperplanes.[7]

For each rational ball, a certain number of cuts is performed: from 1000 in dimension 3 to only 6 in dimension 5 for rational balls obtained from Farey sequences of order 5 and 6 (due to precision issues in Qhull). 95% confidence intervals are depicted for each point (*i.e.* for each rational ball) as error bars, but most of the time too small to be visible on the graphs. Note that this remark suggests that the size of $|P|$ does not depend on the position of $\mathcal{P}$, thus supporting the uniform distribution hypothesis discussed in the previous section.

Analysing these results, we see that $|P|$ seems to behave as f^{α}, with $\alpha < 0$ and decreasing when the dimension increases. This suggests that, in practice, the complexity W of Algorithm 9 is expected to be $O((n+\log f)\cdot \log f)$. Similarly to dimension 2 (see Algorithm 6), this approach is expected to lower down the worst case complexity of the computation of the distance transformation for chamfer norms in dimension n from a logarithmic factor on the size N of the image, to a logarithmic factor on the size f of the rational ball.

7.2 Distance transformation in dimension 2

We evaluate the performance of the separable approach to compute distance transformation for chamfer norms. To efficiently implement predicates leading to the subquadratic algorithm in dimension 2 (Alg. 6 and 7), we store the chamfer norm weighted vectors $\mathcal{M}$ in a random access container sorted counterclockwise to be able to get the mid-vector $\mathbf{v}^{mid}$ in $O(1)$. When implementing Algorithms 6 and 7, few special cases have to be taken into account. For instance, we have to handle situations where a, b or c belong to S in Alg. 6 and 7. Furthermore, Eq. (11) has a solution iff $A_k \neq A_j$. Thanks to the geometrical representation of the binary search process (Fig. 4), such special cases are easy to handle. Fig. 7-(a) illustrates some results on a small domain.

To evaluate experimentally the computational cost given in Theorem 1, we generate m random vectors $(x,y)^T$ with $gcd(x,y)=1$, setting all weights to one. In Fig.7-$(b-c)$, we have considered a 2D domain 2048^2 with 2048 random sites. First, we observe that fixing N, the $\log^2 m$ term is clearly visible in the computational cost of the Voronoi map (single thread curve). Bumps in the single thread curve may be due to memory cache issues. Please note that if we consider classical chamfer norm DT from raster scan (and sub-masks), the computational cost is in $O(m\cdot N^2)$ and thus has a linear behavior in Fig. 7-(c). Since we have a separable algorithm, we can trivially implement it in a multi-thread environment. Hence, on a bi-processor and quad-core (hyper-threading) Intel(R) Xeon(R) cpu (16 threads can run in parallel), we observe a speed-up by a factor 10 (blue curve in Fig. 7-(b)). Please note that on this 2048^2 domain with 2048 sites, Euclidean Voronoi Map (L_2) is obtained in 954.837 milliseconds on a single core and 723.196 msec on 16 cores.

Implementation of all separable algorithms are publicly available in the DGtal library [1].

[7] Python code used to generate Farey sequences and to compute these graphs is available on http://www.gipsa-lab.fr/~isabelle.sivignon/recherches_en.html.

8 Conclusion and Discussion

In this article, we have proposed several generic algorithms to efficiently solve the Voronoi map and distance transformation for a large class of metrics. Focusing on chamfer norms, geometrical interpretation of this generic approach allows us to design a first subquadratic algorithm in dimension 2 to compute the Voronoi map. Thanks to separability, parallel implementation of the distance transformation leads to efficient distance computation.

In higher dimensions, we have shown that all results holds: distance function can be evaluated in $O(n+\log m)$ and the binary search described in VoronoiEdgeWedge can also be extended to n-dimensional chamfer norms.

For the L_2 metric,(additively) weighted voronoi maps, also known as power maps, can be used to solve the reverse distance transformation and medial axis extraction problem using similar separable techniques [11]. A challenging future work would be to extend these results for path-based norms such as chamfer norms.

References

1. DGtal: Digital geometry tools and algorithms library ( ). http://dgtal.org
2. Bajaj, C.L., Pascucci, V.: Splitting a complex of convex polytopes in any dimension. In: Proceedings of the Twelfth Annual Symposium on Computational Geometry, SCG '96, pp. 88–97. ACM (1996)
3. Barber, C.B., Dobkin, D.P., Huhdanpaa, H.: The quickhull algorithm for convex hulls. ACM Trans. Math. Softw. **22**(4), 469–483 (1996). DOI 10.1145/235815.235821. URL www.qhull.org
4. Boca, F.P., Cobeli, C., Zaharescu, A.: Distribution of lattice points visible from the origin. Communications in Mathematical Physics **213**(2), 433–470 (2000)
5. Borgefors, G.: Distance transformations in digital images. Computer Vision, Graphics, and Image Processing (3), 344–371
6. Borgefors, G.: Distance transformations in digital images. Computer Vision, Graphics, and Image Processing **34**(3), 344–371 (1986)

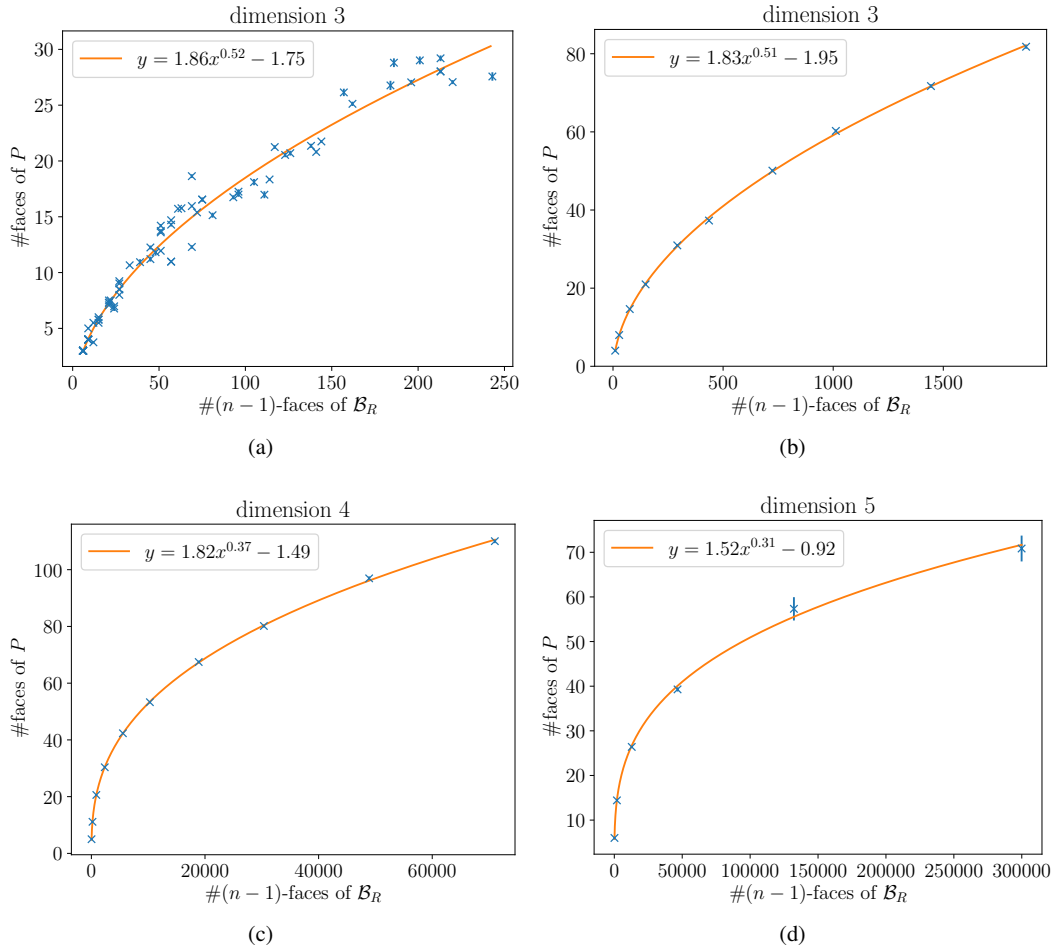

Fig. 6 Number of faces of P with respect to the number of $(n-1)$-faces of $\mathcal{B}_R$ in different settings. In (a), $\mathcal{B}_R$ is a rational ball as computed in [14] in dimension 3. In (b-c), $\mathcal{B}_R$ is defined from a Farey sequence of given order and dimension, taking all the fractions : (b) dimension 3, for orders between 1 and 10, (c) dimension 4 for order between 1 and 10, (d) dimension 5 for orders between 1 and 6. Each point is the mean of a certain number of random cuts (1000 in dimension 3, 500 in dimension 4, 400 in dimension 5 for orders up to 4, and 6 in dimension 5 for orders 5 and 6.).

7. Breu, H., Gil, J., Kirkpatrick, D., Werman, M.: Linear time euclidean distance transform algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence **17**(5), 529–533 (1995)
8. Chazelle, B.: An Optimal Convex Hull Algorithm in Any Fixed Dimension. Discrete \& Computational Geometry **10**, 377–409 (1993)
9. Coeurjolly, D.: Applications of Discrete Geometry and Mathematical Morphology, *LNCS*, vol. 7346, chap. Volumetric Analysis of Digital Objects Using Distance Transformation: Performance Issues and Extensions. Springer-Verlag (2012)
10. Coeurjolly, D.: 2d subquadratic separable distance transformation for path-based norms. In: 18th International Conference on Discrete Geometry for Computer Imagery, pp. 75–87. Springer (2014)
11. Coeurjolly, D., Montanvert, A.: Optimal separable algorithms to compute the reverse euclidean distance transformation and discrete medial axis in arbitrary dimension. IEEE Transactions on PAMI **29**(3), 437–448 (2007)
12. Danielsson, P.E.: Euclidean distance mapping. Computer Graphics and Image Processing **14**, 227–248 (1980)
13. de De, M., van Van, M., Overmars, M., Schwarzkopf, O.: Computational Geometry. Springer-Verlag (2000)
14. Fouard, C., Malandain, G.: 3-D chamfer distances and norms in anisotropic grids. Image and Vision Computing **23**, 143–158 (2005)
15. Hirata, T.: A unified linear-time algorithm for computing distance maps. Information Processing Letters **58**(3), 129–133 (1996)
16. Klette, R., Rosenfeld, A.: Digital Geometry: Geometric Methods for Digital Picture Analysis. Series in Computer Graphics and Geometric Modelin. Morgan Kaufmann (2004)
17. Marklof, J.: Fine-Scale Statistics for the Multidimensional Farey Sequence, pp. 49–57. Springer Berlin Heidelberg (2013)
18. Marklof, J., Strömbergsson, A.: Visibility and directions in quasicrystals. International Mathematics Research Notices **2015**(15), 6588–6617 (2015)
19. Matousek, J., Schwarzkopf, O.: On ray shooting in convex polytopes. Discrete & Computational Geometry **10**, 215–232 (1993)
20. Maurer, C., Qi, R., Raghavan, V.: A Linear Time Algorithm for Computing Exact Euclidean Distance Transforms of Binary Images in Arbitrary Dimensions. IEEE Trans. Pattern Analysis and Machine In **telligence**, 25pp265–270 (2003)
21. Meijster, A., Roerdink, J.B.T.M., Hesselink, W.H.: A general algorithm for computing distance transforms in linear time. In: Mathematical Morphology and its Applications to Image and Signal Processing, pp. 331–340. Kluwer (2000)
22. Mukherjee, J., Das, P.P., Kumarb, M.A., Chatterjib, B.N.: On approximating euclidean metrics by digital distances in 2D and 3D. Pattern Recognition Letters **21**(6–7), 573–582 (2000)

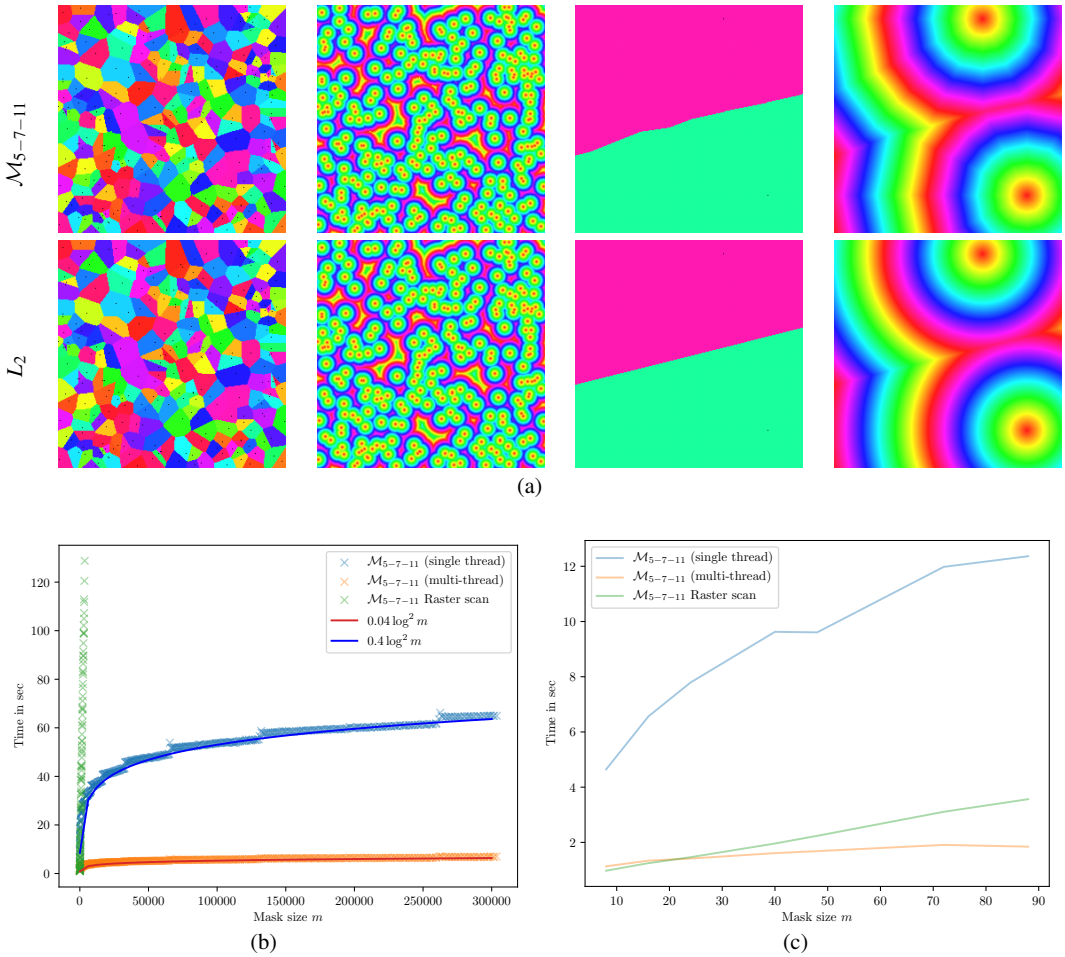

Fig. 7 (a) Separable Voronoi map and distance transformation for the $\mathcal{M}_{5-7-11}$ and L_2 metrics on a 256^2 domain with 256 and 2 random seeds. (b) Experimental evaluation of the subquadratic algorithm when increasing the mask size on a 2048^2 (zoom in (c)).

23. Normand, N., Évenou, P.: Medial axis lookup table and test neighborhood computation for 3d chamfer norms. Pattern Recognition **42**(10), 2288–2296 (2009)
24. Normand, N., Strand, R., Évenou, P.: Digital distances and integer sequences. In: R. González-Díaz, M.J. Jiménez, B. Medrano (eds.) DGCI, *Lecture Notes in Computer Science*, vol. 7749, pp. 169–179. Springer (2013)
25. Ragnemalm, I.: Contour processing distance transforms, pp. 204–211. World Scientific (1990)
26. Rosenfeld, A., Pfaltz, J.: Sequential operations in digital picture processing. Journal of the ACM (JACM) pp. 13pp471–494
27. Rosenfeld, A., Pfaltz, J.: Distance functions on digital pictures. Pattern Recognition **1**, 33–61 (1968)
28. Strand, R.: Distance Functions and Image Processing on Point-Lattices With Focus on the 3D Face- and Body-centered Cubic Grids. Phd thesis, Uppsala Universitet (2008)
29. Thiel, E.: Géométrie des distances de chanfrein. Ph.D. thesis, Aix-Marseille 2 (2001)
30. Verwer, B.J.H., Verbeek, P.W., Dekker, S.T.: An efficient uniform cost algorithm applied to distance transforms. IEEE Transactions on Pattern Analysis and Machine Intelligence **11**(4), 425–429 (1989)