A Novel Algorithm for Distance Transformation on Irregular Isothetic Grids

Antoine Vacavant[1,2], David Coeurjolly[1,3], and Laure Tougne[1,2]

[1] Université de Lyon, CNRS
[2] Université Lyon 2, LIRIS, UMR5205, F-69676, France
[3] Université Lyon 1, LIRIS, UMR5205, F-69622, France
{antoine.vacavant,david.coeurjolly,laure.tougne}@liris.cnrs.fr

Abstract. In this report, we propose a new definition of the E^2DT (Squared Euclidean Distance Transformation) on irregular isothetic grids. We describe a new separable algorithm to compute this transformation on every grids, which is independent of the background representation. We show that our proposal is able to efficiently handle various kind of classical irregular two-dimensional grids in imagery, and that it can be easily extended to higher dimensions.

1 Introduction

The representation and the manipulation of large-scale data sets by irregular grids is a today scientific challenge for many applications. For example, computing the solutions of partial differential equations can be significantly improved by a multi-grid approach [14] or octrees [15]. Here, we are interested in the notion of discrete distance on this kind of grids, which is a very important concept in image analysis and description on classical regular grids [10, 16].The Distance Transformation (DT) of a binary image consists in labeling each point of a discrete object $\mathcal{E}$ (*i.e.* foreground) with its shortest distance to the complement of $\mathcal{E}$ (*i.e.* background). This process is widely studied and developed on regular grids (see for example [5]). Some specific extensions of the DT to non-regular grids also exist, such as rectangular grids [1, 19], quadtrees [18, 21], *etc.*

This report deals with generalizing the DT computation on *irregular isothetic grids* (or $\mathbb{I}$-grid for short) in two dimensions (2-D). The proposed method is easily extensible to higher dimensions. In 2-D, this kind of grids is defined by rectangular cells whose edges are aligned along the two axis. The size and position of those non-overlapping cells are defined without constraint (see next section for a formal definition). The quadtree decomposition and the RLE (Run Length Encoding) grouping schemes are examples of classical techniques in imagery which induce an $\mathbb{I}$-grid. Here, we focus our interest on generalizing techniques that compute the E^2DT (squared Euclidean DT) of a d-dimensional (d-D) binary image [3, 11, 17]. Many of those methodologies can be linked to the computation of a discrete Voronoi diagram of the background pixels [7, 3]. In a previous work [20], we introduced a definition of E^2DT on $\mathbb{I}$-grids based on the background

cells centers. After showing the link between this process and the computation of the Voronoi diagram of those background points, we proposed two completing algorithms to handle various configurations of $\mathbb{I}$-grids. Unfortunately, we noticed that they suffer from non-optimal time complexity. Moreover, the result of this transformation strongly depends on the representation of the background. In this document, we present a new way to compute the E^2DT on $\mathbb{I}$-grids, based on the border of the background cells. We show that this transformation is thus independent of the background encoding. We describe an algorithm based on the work of R. Maurer *et al.* [11] to efficiently compute this new transformation. We finally show the interest of our contribution by comparing it with the techniques we proposed in [20]. We mainly consider the case of 2-D $\mathbb{I}$-grids in this report, and we will shortly show that our contribution is easily extensible to the d-D case.

The next section of this report presents our new definition of E^2DT on $\mathbb{I}$-grids, and its relation with the computation of a Voronoi diagram of segments. In Section 3, we describe the extension of the algorithm of R. Maurer *et al.* [11] to compute this transformation in 2-D but extensible to higher dimensions. We also proove the correctness of our proposal, in the similar way as T. Hirata showed that his method computes exactely the E^2DT of a binary image [8]. We finally show in Section 4 experimental results to compare our new contribution with the techniques we developed in [20] in terms of speed and time complexity.

2 Distance Transformations on $\mathbb{I}$-grids and Voronoi Diagrams

In this section, we first introduce the concept of irregular isothetic grids ($\mathbb{I}$-grids), with the following definition [2]:

Definition 1 (2-D $\mathbb{I}$-grid). *Let $I \subset \mathbb{R}^2$ be a closed rectangular support. A 2-D $\mathbb{I}$-grid $\mathbb{I}$ is a tiling of I with non overlapping rectangular cells which edges are parallel to the X and Y axis. The position (x_R, y_R) and the size (l_R^x, l_R^y) of a cell R in $\mathbb{I}$ are given without constraint:*

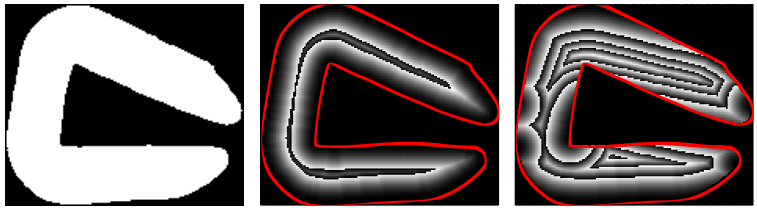
$$(x_R, y_R) \in \mathbb{R}^2,\ (l_R^x, l_R^y) \in \mathbb{R}^2. \tag{1}$$

We consider this definition of 2-D $\mathbb{I}$-grids for the rest of the report, and we will shortly show that our contribution is easily extensible to the d-D case. In our framework, we consider *labeled* $\mathbb{I}$-grids, *i.e.* each cell of the grid has a *foreground* or *background* label (its value is respectively "0" or "1" for example). For an $\mathbb{I}$-grid $\mathbb{I}$, we denote by $\mathbb{I}_F$ and $\mathbb{I}_B$ the sets of foreground and background cells.

We can first consider that the distance between two cells R and R' is the distance between their centers. If we denote $p = (x_R, y_R)$ and $p' = (x_{R'}, y_{R'})$ these points, and $d_e^2(p, p')$ the squared Euclidean distance between them, the $\mathbb{I}$-CDT (Center-based DT on $\mathbb{I}$-grids) of a cell R is defined as follows [20]:

$$\mathbb{I}\text{–CDT}(R) = \min_{R'} \left\{ d_e^2(p, p');\ R' \in \mathbb{I}_B \right\}, \tag{2}$$

and is exactly the E^2DT if we consider $\mathbb{I}$ as a regular (square or rectangular) discrete grid. However, this model is strongly dependent of the background representation. In Figure 1, we present an example of the computation of the $\mathbb{I}$-CDT of two $\mathbb{I}$-grids where only the background cells differ. Since this definition is based on the cells centers position, the $\mathbb{I}$-CDT do not lead to the same distance map according to the background coding.



(a) (b) (c)

Fig. 1. The result of the $\mathbb{I}$-CDT of the complete regular grid (b) computed from the binary image (a) and an $\mathbb{I}$-grid where the foreground is regular and the background is encoded with a RLE scheme along Y (c). The distance value d of a cell is represented with a grey level $c = d \mod 255$. The contour of the object (background/foreground frontier) is recalled in each distance map (b) and (c) with a smooth curve

We now choose to extend the definition of the E^2DT by considering the shortest distance between a foreground cell and the background/foreground boundary of the $\mathbb{I}$-grid. We suppose here that the intersection between two adjacent cells is a segment of dimension one (*i.e.* they respect the e-adjacency relation [2]). Let $\mathcal{S}$ be the set of segments at the interface between two adjacent cells $R \in \mathbb{I}_F$ and $R' \in \mathbb{I}_B$.The $\mathbb{I}$-BDT (Border-based DT on $\mathbb{I}$-grids) is then defined as follows:

$$\mathbb{I}\text{−BDT}(R) = \min_{s} \left\{ d_e^2(p, s);\ s \in \mathcal{S} \right\}. \tag{3}$$

Contrary to the $\mathbb{I}$-CDT given in Equation 2, this process does not take into account the representation of the background. We can draw a parallel between those extensions of the E^2DT on $\mathbb{I}$-grids and the computation of a Voronoi diagram (VD). More precisely, as in the regular case, the $\mathbb{I}$-CDT can be linked with the VD of the background cells centers (*i.e. Voronoi sites* or *VD sites*) [20]. The VD of a set of points $\mathcal{P} = \{p_i\}$ is a tiling of the plane into *Voronoi cells* (or *VD cells*) $\{C_{p_i}\}$ [4]. If we now consider the background/foreground frontier to compute the $\mathbb{I}$-BDT, our definition implies that we compute a VD of segments, and not a classical VD of points (see Figure 2 for an example of these diagrams computed on a simple $\mathbb{I}$-grid). Hence, a simple approach to compute the $\mathbb{I}$-CDT is to pre-compute the complete VD of the background points [20], and to locate foreground points in the VD. In this case, the $\mathbb{I}$-CDT computation has a $\mathcal{O}(n \log n_B)$ time complexity, where n is the total number of cells, and n_B is the number of background cells. This technique is obviously not computationally

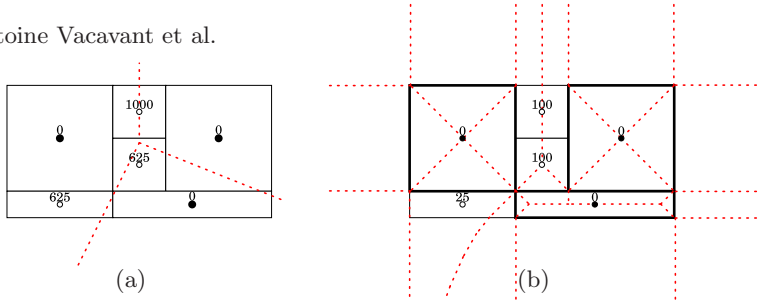


Fig. 2. Example of the VD of the background points to obtain the $\mathbb{I}$-CDT of a simple $\mathbb{I}$-grid (a). Background cell centers are depicted in black, and foreground ones in white. Distance values are presented above the cells centers. From the same grid, the computation of the $\mathbb{I}$-BDT (b) implies that we consider the VD of the segments belonging to the background/foreground frontier

efficient for every grids, and not adapted to dense grids [20]. To compute the $\mathbb{I}$-BDT, a similar synopsis can be drawn, where the computation of the VD of segments can be handled in $\mathcal{O}(n_S \log^2 n_S)$, where $n_S = 4n_B$ is the total number of segments belonging to the background/foreground frontier [9]. The extension of those transformation to d-D $\mathbb{I}$-grids is a hard work. A VD can be computed in d-D with a $\mathcal{O}(n_B \log n_B + n_B^{\lceil d/2 \rceil})$ time complexity (thanks to a gift-wrapping approach [4] for example). However, localizing a point in the VD is an arduous task, and an additional structure like subdivision grids [13] should be constructed to handle this operation.

In this section, we have presented a new extension of the E^{2}DT on $\mathbb{I}$-grids, based on the background/foreground frontier of the grid. Hence, we have shown that building the entire VD to obtain the $\mathbb{I}$-BDT is neither computationally efficient for every $\mathbb{I}$-grids, nor easily extensible to higher dimensions. We now propose a separable algorithm to compute the $\mathbb{I}$-BDT that can be extented to the d-D case.

3 A New Separable Algorithm to Compute the $\mathbb{I}$-BDT

3.1 R. Maurer *et al.* E^{2}DT Algorithm on Regular Grids

The main idea of this separable method [5, 11] is that the intersection between the complete VD of background pixels (*i.e.* sites) and a line of the grid can be easily computed, then simplified. Indeed, for a row j, the VD sites can be "deleted" by respecting three remarks mainly based on the monotonicity of d_e distance [11]: (1) if we consider the line $l : y = j,\ j \in \mathbb{Z}$, then we only have to keep the nearest VD sites from l (Figure 3-b), (2) those sites can be ordered along the X axis, which means that it is not necessary to keep the complete VD, but only its intersection with the line l, (3) a VD site may be *hidden by* two neighbour sites, and thus not considered anymore (Figure 3-c). In this report, we show the extension of this technique on $\mathbb{I}$-grids by adapting these properties on these grids.

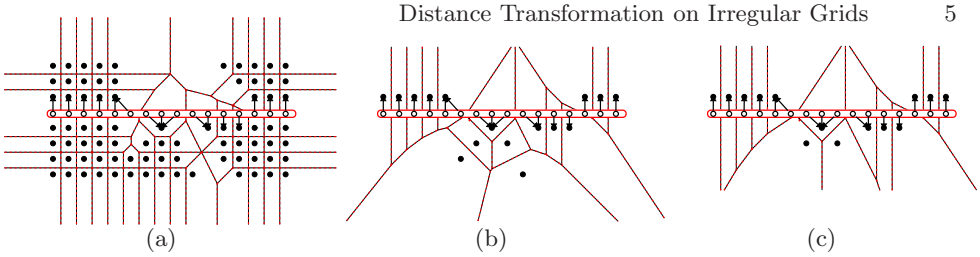
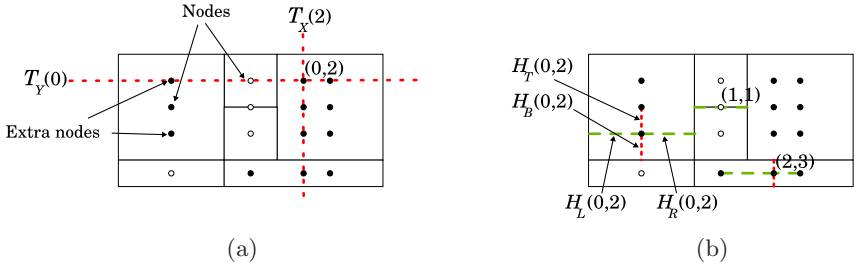


Fig. 3. In a regular grid, when we treat a row l with R. Maurer *et al.* algorithm, we consider the VD of background nodes like in (a). We only keep the nearest VD sites of l (b). We obtain (c) by deleting sites which associated VD cells intersect the l row. Arrows indicate the associated VD site of each foreground pixel of this row

3.2 Separable Computation of the $\mathbb{I}$-BDT

To develop a separable process on $\mathbb{I}$-grids, we use a similar structure as the *irregular matrix* $\mathbf{A}$ associated to a labeled $\mathbb{I}$-grid $\mathbb{I}$ introduced in [20]. This data structure aims to organize the cells of the grid along the X and Y axis by adding virtual cells centers (see Figure 4 for an example). We used it in our previous work to adapt the separable E^{2}DT algorithm of T. Saito *et al.* [17], devoted to compute the $\mathbb{I}$-CDT on $\mathbb{I}$-grids. The irregular matrix contains as



(a) (b)

Fig. 4. Construction of the irregular matrix associated to the simple $\mathbb{I}$-grid illustrated in Figure 2. In (a), the extra node $\mathbf{A}(0,2)$ of the matrix is depicted at the intersection of the dotted lines, and nodes have the same value as the cell containing them. For the $\mathbb{I}$-BDT, we also need the shortest distance between a node in respect to its neighbour nodes and the border of the cell containing it (b). Examples of values for border attributes are also given along X (dashed lines) and Y (dotted lines). For instance, we have $H_T(1,1) = H_B(1,1) = 0$ since this node coincides with a cell horizontal border. We can also notice that $H_L(2,3) \neq H_R(2,3)$

many columns (respectively rows) as X-coordinates (Y-coordinates) in the grid. These coordinates are stored in two tables T_X and T_Y (and we denote $n_1 = |T_X|$ and $n_2 = |T_Y|$). At the intersection of two X and Y coordinates, a node in $\mathbf{A}$ has the same value as the cell containing it, and may represent the cell center or not (*i.e.* this is an *extra node*, see Figure 4-a). The extra nodes are used to propagate the distance values through the irregular matrix and then compute a correct distance value for each cell center. To apply the $\mathbb{I}$-BDT on this data

structure, we also have to take into account the border and the size of the treated cells. We thus add for each node $\mathbf{A}(i,j)$ *border attributes* along X and Y axis that permit to propagate the distance values to cell borders through $\mathbf{A}$. For a node $\mathbf{A}(i,j)$ contained by the cell R in $\mathbb{I}$, we denote $H_L(i,j)$ and $H_R(i,j)$ attributes that represent respectively the minimum between the distance to the left (right) border of R along X, and the distance to the neighbour node at the left (right) position in $\mathbf{A}$. In the same manner, we define $H_T(i,j)$ and $H_B(i,j)$ in respect to the top (bottom) border of R and neighbour nodes at the top (bottom) position in $\mathbf{A}$ (see Figure 4-b). Building the irregular matrix of an $\mathbb{I}$-grid $\mathbb{I}$ can be handled in $\mathcal{O}(n_1 n_2)$ time complexity. More precisely, we first scan all the cells of $\mathbb{I}$ to know the n_1 columns and the n_2 rows of $\mathbf{A}$. Then, we consider each node of $\mathbf{A}$ and we assign its background or foreground value and its border attributes.

Equation 3 can now be adapted on the irregular matrix with a minimization process along the two axis (a proof is given in Appendix A of this report):

Proposition 1 (Separable $\mathbb{I}$-BDT). *Let $\mathbf{A}$ be the associated irregular matrix of the 2-D $\mathbb{I}$-grid $\mathbb{I}$. Then the $\mathbb{I}$-BDT of $\mathbb{I}$ can be decomposed in two separable processes, and consists in computing the matrix $\mathbf{B}$ and $\mathbf{C}$ as follows:*

$$\mathbf{B}(i,j) = \min_x \Big\{ \min\big(|T_X(i) - T_X(x) - H_R(x,j)|, |T_X(x) - T_X(i) - H_L(x,j)|\big) \; ; \; x \in \{0,...,n_1-1\}, \; \mathbf{A}(x,j) = 0 \Big\},$$

$$\mathbf{C}(i,j) = \min_y \big\{ \mathcal{G}_y(j); \; y \in \{0,\ldots,n_2-1\} \big\},$$

where $\mathcal{G}_y(j)$ is the flattened parabola given by:

$$\mathcal{G}_y(j) = \begin{cases} \mathbf{B}(i,y)^2 + (T_Y(j) - T_Y(y) - H_T(i,y))^2 & if \; T_Y(j) - T_Y(y) > H_T(i,y) \\ \mathbf{B}(i,y)^2 + (T_Y(y) - T_Y(j) - H_B(i,y))^2 & if \; T_Y(y) - T_Y(j) > H_B(i,y) \\ \mathbf{B}(i,y)^2 & else. \end{cases}$$

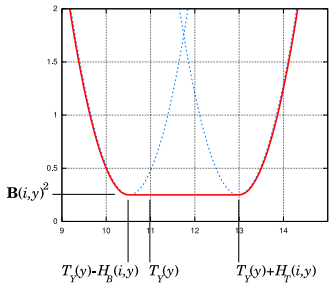


Fig. 5. Important features of a flattened parabola that is indeed the composition of two half-parabolas (in dotted lines) and a constant function

This separable minimization process also permits to compute the $\mathbb{I}$-CDT (Equation 2), if we assign all border attributes to zero. In this case, this transformation

is equivalent to a minimization process of a 2-D quadratic form [20], and we consider a set of classical parabolas along the Y axis, as in the regular case [17]. Here, a flattened parabola (see also Figure 5) is an *ad-hoc* function composed by two half-parabolas and a constant function. It represents a constant distance value from a node $\mathbf{A}(i,j)$ to its neighbour nodes and cell borders, and then increases as a classical quadratic function beyond those limits. The computation of matrix $\mathbf{C}$ in Proposition 1 thus consists in computing the lower enveloppe of a set of flattened parabolas. This operation is possible since these functions are monotone (composition of monotone functions), and there exist a single intersection or an infinity of intersections between two flattened parabolas. In the regular case, E^{2}DT algorithms based on a 2-D quadratic form minimization [8, 12, 17] are linear and correct, since the intersection between two parabolas $\mathcal{F}_\alpha(y)$ and $\mathcal{F}_\beta(y)$ is clearly defined:

Proposition 2 (Intersection between two classical parabolas [8]). *Let $\mathcal{F}_\alpha : \mathbb{R} \to \mathbb{R}, y \to g_\alpha^2 + (\alpha - y)^2$ and $\mathcal{F}_\beta : \mathbb{R} \to \mathbb{R}, y \to g_\beta^2 + (\beta - y)^2$ be two parabolas. The number of intersections is either one, or an infinity if they are coincident, i.e. when $g_\alpha = g_\beta$ and $\alpha = \beta$.*

This property is verified because parabolas are totally monotone functions [8]. It implies that we can compute the lower enveloppe of a set of parabolas since we are able to order them. To proove that our algorithm is correct on $\mathbb{I}$-grids, we thus have to verify that this property is preserved. The following lemma enounce the conditions so that two flattened parabolas intersect at a single point or an infinity of points:

Lemma 1 (Intersection between two flattened parabolas). *Let $\mathcal{G}_u$ and $\mathcal{G}_v$ be two flattened parabolas given by:*

$$\mathcal{G}_u : \mathbb{R} \to \mathbb{R}$$
$$y \to \mathcal{G}_u(y) = \begin{cases} g_u^2 + (u - y - lu_1)^2 & if\ u - y > lu_1 \\ g_u^2 + (y - u - lu_2)^2 & if\ y - u > lu_2 \\ g_v^2 & else \end{cases}$$

and

$$\mathcal{G}_v : \mathbb{R} \to \mathbb{R}$$
$$y \to \mathcal{G}_v(y) = \begin{cases} g_v^2 + (v - y - lv_1)^2 & if\ v - y > lv_1 \\ g_v^2 + (y - v - lv_2)^2 & if\ y - v > lv_2 \\ g_v^2 & else \end{cases}$$

where:

- *g_u, g_v, lu_1, lu_2, lv_1, lv_2, u and v are positive or null real numbers;*
- *if $u \geq v \geq 0$, then $u - lu_1 \geq v + lv_2$;*
- *if $0 \leq u \leq v$, then $u + lu_2 \leq v - lu_1$.*

The number of intersections between these parabolas is either one, either an infinity.

Notice that the elements $\mathbf{B}(i,y)$, $H_T(i,y)$, $H_B(i,y)$ and $T_Y(y)$ of $\mathcal{G}_y$ in Proposition 1 respectively correspond to g_u, lu_1, lu_2, and u of $\mathcal{G}_u$. They also respect the conditions given at the end of Lemma 1. For example, let $\mathcal{G}_{y_1}$ and $\mathcal{G}_{y_2}$ be two flattened parabolas such that $y_1 \geq y_2$, we can easily verify that $y_1 - H_B(i,y_1) \geq y_2 + H_T(i,y_2)$ (see Figure 4 and Figure 5). The proof of this lemma and a way to compute a valid intersection point are given in Appendix B of this report.

3.3 Adaptation of R. Maurer *et al.* $\mathbf{E^2DT}$ Algorithm on $\mathbb{I}$-grids

The first stage of our method (first step of Algorithm 1) consists in scanning along X and in initializing the distance of each node of the irregular matrix. This is indeed a double linear scan for each row. Notice the use of H_R and H_L attributes to propagate the distance to cells borders. At the end of this stage, each node store a squared distance (line 14). In the second part of our algorithm, we call the `Voronoi_IBDT` function (Algorithm 2) to build a partial VD intersected with each column i. As in the original algorithm [11], we use two stacks storing real successive coordinates of treated sites (h), and their squared distance (g). The first loop of this function (line 3) corresponds to the deletion of hidden sites, thanks to the `hidden_by()` predicate (Algorithm 3). In Algo-

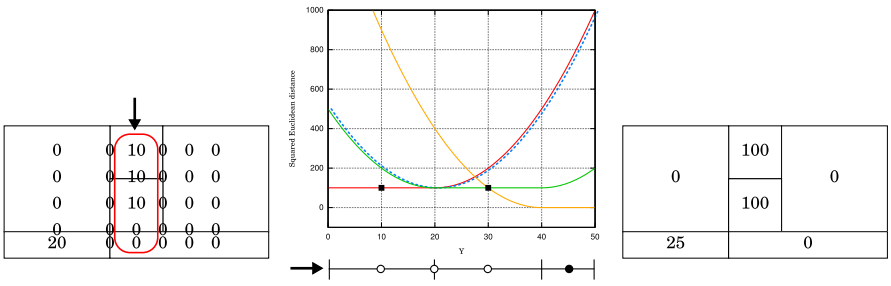


Fig. 6. For the column chosen in (a), the last phase of Algorithm 2 consists in considering the lower enveloppe of a set of flattened parabolas (b). At the bottom of this plot, background and foreground nodes are represented by black and white circles at the corresponding Y-coordinate. Cell borders are also represented (vertical dashes). Black squares represent where the cell centers are located along Y axis, and the associated $\mathbb{I}$-BDT value. We give in (c) the obtained $\mathbb{I}$-BDT for this $\mathbb{I}$-grid

rithm 2, we also use two additional stacks, denoted f_T and f_B to store the border attributes and update them (line 6). Thanks to these stacks, the second stage of our algorithm is achieved in linear time. By testing the value of l (line 7), we know if we have to scan again the stacks and to update the distance values of the nodes. Finally, we linearly scan the stacks to find the nearest border of the

Algorithm 1: Separable computation of the $\mathbb{I}$-BDT inspired from [11].

input : the labeled $\mathbb{I}$-grid $\mathbb{I}$.
output: the $\mathbb{I}$-BDT of $\mathbb{I}$, stored in the irregular matrix $\mathbf{C}$.

build the irregular matrix $\mathbf{A}$ associated to $\mathbb{I}$;
for $j = 0$ **to** $n_2 - 1$ **do** {*First stage along* X}
  if $\mathbf{A}(0, j) = 0$ **then** $\mathbf{B}(0, j) \leftarrow 0$;
  else $\mathbf{B}(0, j) \leftarrow \infty$;
  for $i = 1$ **to** $n_1 - 1$ **do**
    if $\mathbf{A}(i, j) = 0$ **then** $\mathbf{B}(i, j) \leftarrow 0$;
    else
      if $\mathbf{B}(i-1, j) = 0$ **then** $\mathbf{B}(i, j) \leftarrow T_X(i) - T_X(i-1) - H_R(i-1, j)$;
      else $\mathbf{B}(i, j) \leftarrow T_X(i) - T_X(i-1) + \mathbf{B}(i-1, j)$;
  for $i = n_1 - 2$ **to** 0 **do**
    if $\mathbf{B}(i+1, j) < \mathbf{B}(i, j)$ **then**
      if $\mathbf{B}(i+1, j) = 0$ **then** $\mathbf{B}(i, j) \leftarrow T_X(i+1) - T_X(i) - H_L(i, j)$;
      else $\mathbf{B}(i, j) \leftarrow T_X(i+1) - T_X(i) + \mathbf{B}(i-1, j)$;
    $\mathbf{B}(i, j) \leftarrow \mathbf{B}(i, j)^2$;
for $i = 0$ **to** $n_1 - 1$ **do** {*Second stage along* Y}
  `Voronoi_IBDT`(i);

Algorithm 2: Function `Voronoi_IBDT()` to build a partial VD along Y.

input : the column i of the irregular matrix $\mathbf{B}$.
output: the $\mathbb{I}$-BDT of each node of the column i stored in the matrix $\mathbf{C}$.

$l \leftarrow 0$, $g \leftarrow \emptyset$, $h \leftarrow \emptyset$, $f_T \leftarrow \emptyset$, $f_B \leftarrow \emptyset$;
for $j = 0$ **to** $n_2 - 1$ **do**
  if $\mathbf{B}(i, j) \neq \infty$ **then**
    while $l \geq 2 \wedge$ `hidden_by`$\big(g[l-1], g[l], \mathbf{B}(i, j), h[l-1], h[l], T_Y(j)\big)$ **do**
      $l \leftarrow l - 1$;
    $l \leftarrow l + 1$, $g[l] \leftarrow \mathbf{A}(i, j)$, $h[l] \leftarrow T_Y(j)$, $f_T[l] \leftarrow H_T(i, j)$, $f_B \leftarrow H_B(i, j)$;
if $(n_s \leftarrow l) = 0$ **then return**;
$l \leftarrow 1$;
for $j = 0$ **à** $n_2 - 1$ **do**
  while $l < n_s \ \wedge \ \mathcal{G}_l(j) > \mathcal{G}_{l+1}(j)$ **do**
    $l \leftarrow l + 1$;
  $\mathbf{A}(i, j) \leftarrow \mathcal{G}_l(j)$;

Algorithm 3: Predicate `hidden_by()`.

input : Y-coordinates of three points in $\mathbb{R}^2$ denoted u_y, v_y, w_y, and their squared distance to the line $L : y = r$ denoted $d_e^2(u, L)$, $d_e^2(v, L)$, $d_e^2(w, L)$.
output: is v hidden by u and w ?

$a \leftarrow v_y - u_y$, $b \leftarrow w_y - v_y$, $c \leftarrow a + b$;
return $c \times d_e^2(v, L) - b \times d_e^2(u, L) - a \times d_e^2(w, L) - abc > 0$;

$\mathbf{A}(i,j)$ current node (line 10), and this step indeed consists in considering the lower enveloppe of a set of flattened parabolas $\{\mathcal{G}_l\}$, given by (see also Figure 6):

$$\mathcal{G}_l(j) = \begin{cases} g[l] + (T_Y(j) - h[l] - f_T[l])^2 \; if \; T_Y(j) - h[l] > f_T[l] \\ g[l] + (h[l] - T_Y(j) - f_B[l])^2 \; if \; T_Y(j) - h[l] > f_B[l] \\ \mathbf{B}(i,y)^2 \qquad\qquad\qquad\qquad\qquad else, \end{cases} \tag{4}$$

which is the adaptation of Proposition 1 with the stacks g, h, f_T and f_B. We thus can enounce the following lemma, directly obtained from Proposition 1 and Lemma 1:

Lemma 2. *Algorithm 1, devoted to compute a separable $\mathbb{I}$-BDT, is correct. More precisely, (1) the first phase consists in computing a cell border based mono-dimensional distance transformation along X axis, and (2) the minimization process along Y axis computes the lower enveloppe of a set of flattened parabolas, giving a correct distance map based on cells borders.*

As a consequence, we have described a linear and correct $\mathbb{I}$-BDT algorithm in respect to the associated irregular matrix size, *i.e.* in $\mathcal{O}(n_1 n_2)$ time complexity. Our new contribution is easily extensible to higher dimensions: we still realize the first step as an initialization phase, and for each dimension $d > 1$, we combine results obtained in dimension d-1. If we consider a labeled d-D $\mathbb{I}$-grid, which associated irregular matrix size is $n_1 \times \cdots \times n_d$, the time complexity of our algorithm is thus in $\mathcal{O}(n_1 \times \cdots \times n_d)$. In the next section, we present experiments to show the interest of the $\mathbb{I}$-BDT, and to point out that our algorithm is a very efficient approach to compute both $\mathbb{I}$-CDT and $\mathbb{I}$-BDT of various classical $\mathbb{I}$-grids.

4 Experimental Results

We first propose to present the result of our $\mathbb{I}$-BDT algorithm for the binary image depicted at the beginning of this report in Figure 1, digitized in various classical $\mathbb{I}$-grids in imagery. Table 1 illustrates $\mathbb{I}$-BDT elevation maps where the background and the foreground of the original image are independently represented with a regular square grid ($\mathbb{D}$), a quadtree ($q\mathbb{T}$) and a RLE along Y ($\mathbb{L}$). We can notice in this figure that the result of the $\mathbb{I}$-BDT is independent of the representation of the background. The distance values in the foreground region are thus the same in the three elevation maps of a given column.

We now focus our interest on the execution time of our new algorithm, in respect to our previous work [20]. We consider the three algorithms presented in Table 2. The first algorithm represents the simple approach we discussed in Section 2, which is hardly extensible to d-D treatments, and $\mathbb{I}$-BDT computation. Algorithm 2 is described in [20] and is inspired from the quadradic form minimization scheme of T. Saito *et al.* [17]. Thanks to the flattened parabola definition, we can extend this algorithm to $\mathbb{I}$-BDT, but with a non-optimal time complexity, in respect to the irregular matrix size. Algorithm 3 is our new separable transformation inspired from [11]. In Figure 7, we present the three chosen images for our experiments, and in Table 3, we show the execution times for

(a) *noise* (b) *lena* (c) *canon*

Fig. 7. For the images *noise* (100x100), *lena* (208x220), *canon* (512x512) we consider in our experiments the associated $\mathbb{l}$-grids (regular grid, quadtree decomposition and RLE along Y scheme)

Table 1. Elevation maps representing the $\mathbb{l}$-BDT of each $\mathbb{l}$-grid. X and Y are the axis of the image, and Z corresponds to the distance value. The foreground (columns) and the background (rows) of the image are digitized independently. The color palette is depicted on the right of the figure

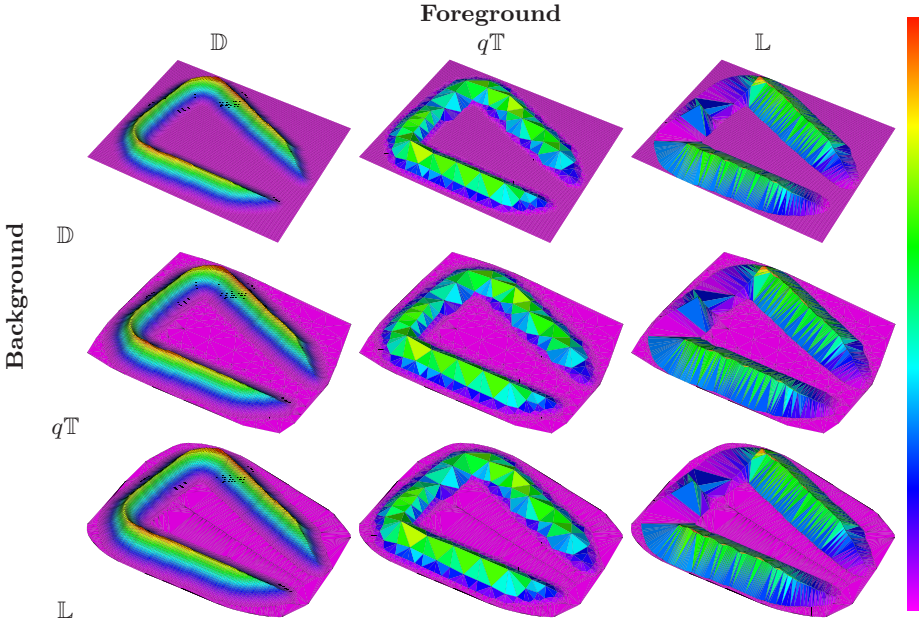


Table 2. The three compared algorithms, and their associated time and space complexities. We also check if an algorithm is extensible to d-D $\mathbb{l}$-grids and what kind of transformation it can perform ($\mathbb{l}$-CDT, $\mathbb{l}$-BDT)

Id.	Algorithm	Time	Space	d-D	$\mathbb{l}$-CDT	$\mathbb{l}$-BDT
1	Complete VD [20]	$\mathcal{O}(n \log n_B)$	$\mathcal{O}(n)$		✓	
2	From T. Saito *et al.* [17, 20]	$\mathcal{O}(n_1 n_2 \log n_2)$	$\mathcal{O}(n_1 n_2)$	✓	✓	✓
3	From R. Maurer *et al.* [11]	$\mathcal{O}(n_1 n_2)$	$\mathcal{O}(n_1 n_2)$	✓	✓	✓

these three algorithms, for $\mathbb{I}$-grids built from three binary images. We have performed those experiments on a mobile workstation with a 2.2 Ghz Intel Centrino Duo processor, and 2 Gb RAM. We can first notice that our new contribution

Table 3. We present execution times (in seconds) for each algorithm for the $\mathbb{I}$-CDT (a) and the $\mathbb{I}$-BDT (b) and for each $\mathbb{I}$-grid. Number inside parenthesis in (b) are the increasing rate in % between $\mathbb{I}$-CDT and $\mathbb{I}$-BDT execution times

(a) $\mathbb{I}$-CDT

Image	**Algorithm 1**			**Image**	**Algorithm 2**			**Image**	**Algorithm 3**		
	$\mathbb{D}$	$q\mathbb{T}$	$\mathbb{L}$		$\mathbb{D}$	$q\mathbb{T}$	$\mathbb{L}$		$\mathbb{D}$	$q\mathbb{T}$	$\mathbb{L}$
noise	0.255	0.104	0.053	*noise*	0.037	0.077	0.044	*noise*	0.046	0.065	0.038
lena	1.413	0.145	0.081	*lena*	0.192	0.376	0.245	*lena*	0.185	0.166	0.135
canon	36.236	0.273	0.234	*canon*	1.678	1.322	1.316	*canon*	1.134	0.485	0.585

(b) $\mathbb{I}$-BDT

Image	**Algorithm 2**			**Image**	**Algorithm 3**		
	$\mathbb{D}$	$q\mathbb{T}$	$\mathbb{L}$		$\mathbb{D}$	$q\mathbb{T}$	$\mathbb{L}$
noise	0.047 (27)	0.100 (29)	0.054 (23)	*noise*	0.065 (42)	0.085 (31)	0.049 (29)
lena	0.256 (33)	0.491 (31)	0.320 (31)	*lena*	0.258 (40)	0.209 (26)	0.170 (26)
canon	2.248 (34)	2.107 (59)	2.020 (53)	*canon*	1.507 (33)	0.563 (16)	0.718 (23)

gives good results for $\mathbb{I}$-CDT. Indeed, Algorithm 3 is the fastest one for dense $\mathbb{I}$-grids (regular grids or $\mathbb{L}$ for *noise*), and is very competitive for sparse $\mathbb{I}$-grids (*e.g.* near one half second for $q\mathbb{T}$ and $\mathbb{L}$ based on image *canon*). In the latter case (*canon*), Algorithm 1 is slightly faster than our approach, but we recall that it is hardly extensible to higher dimensions. Algorithm 2, which we developped in our previous work [20], was interesting for dense grids, and is now overtaken by Algorithm 3. For the $\mathbb{I}$-BDT, Algorithm 2 and 3 suffer from an execution time increase, mainly due to the integration of the complex flattened parabola. But our contribution remains as the fastest algorithm, and the time increasing rate is moderate for the tested grids. Large sparse $\mathbb{I}$-grids like $q\mathbb{T}$ and $\mathbb{L}$ based on *canon* are still handled in less than one second.

5 Conclusion and Future Works

In this report, we have proposed a new extension of the E^2DT on $\mathbb{I}$-grids based on the background/foreground frontier, and independent of the background representation. We have also developed a new separable algorithm inspired from [11] that is able to efficiently compute the $\mathbb{I}$-BDT thanks to the irregular matrix structure. We have finally shown that our new contribution is adaptable to various configurations of $\mathbb{I}$-grids, with competitive execution time and complexities, in comparison with our previous proposals.

As a future work, we would like to develop the extension of the discrete medial axis transform [3] on $\mathbb{I}$-grids with our new definition. This would permit to propose a centred simple form of a binary objet, independently of the representation of the background, and surely extensible to higher dimensions. As in this document, we aim to propose a linear algorithm to construct an irregular medial axis, in respect to the irregular matrix size.

References

1. Chehadeh, Y., Coquin, D., Bolon, P.: A Skeletonization Algorithm Using Chamfer Distance Transformation Adapted to Rectangular Grids. In *13th International Conference on Pattern Recognition (ICPR'96)*, **2** (1996) 131–135.
2. Coeurjolly, D., Zerarga, L.: Supercover Model, Digital Straight Line Recognition and Curve Reconstruction on the Irregular Isothetic Grids. In *Computer and Graphics*, 30(1):46–53, 2006.
3. Coeurjolly, D., Montanvert, A.: Optimal Separable Algorithms to Compute the Reverse Euclidean Distance Transformation and Discrete Medial Axis in Arbitrary Dimension. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**(3) (2007) 437–448.
4. de Berg, M., van Kreveld, M., Overmars, M. and Schwarzkopf, O.: Computational Geometry: Algorithms and Applications. Springer (2000).
5. Fabbri, R., Costa, L. D. F., Torelli, J. C. and Bruno, O. M.: 2D Euclidean Distance Transform Algorithms: A Comparative Survey. In *ACM Computing surveys*, **40**(1) (2008) 1–44.
6. Fouard, C., Malandain, G.: 3-D Chamfer Distances and Norms in Anisotropic Grids. In *Image and Vision Computing*, **23**(2) (2005) 143–158.
7. Hesselink, W.H., Visser, M., Roerdink, J.B.T.M.: Euclidean Skeletons of 3D Data Sets in Linear Time by the Integer Medial Axis Transform. In *7th International Symposium on Mathematical Morphology*, (2005) 259–268.
8. Hirata, T.: A Unified Linear-Time Algorithm for Computing Distance Maps. In *Information Processing Letters*, **58**(3)(1996) 129–133.
9. Karavelas, M.I.: Voronoi diagrams in CGAL. In *22nd European Workshop on Computational Geometry (EWCG06)*, (2006) 229–232.
10. Klette, R. and Rosenfeld, A.: *Digital Geometry: Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann Publishers Inc. (2004).
11. Maurer, C. R., Qi, R., Raghavan, V.: A Linear Time Algorithm for Computing Exact Euclidean Distance Transforms of Binary Images in Arbitrary Dimensions. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(2) (2003) 265–270.
12. 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*, (2000) 331–340.
13. Park, S. H., Lee, S. S. and Kim, J. H.: The Delaunay Triangulation by Grid Subdivision. In *Computational Science and Its Applications (ICCSA 2005)*, (2005) 1033–1042.
14. Plewa, T., Linde, T. and Weirs, V., editors: Adaptive Mesh Refinement - Theory and Applications. Proceedings of the *Chicago Workshop on Adaptive Mesh Refinement in Computational Science and Engineering* **41** (2005).

15. Popinet, S.: Gerris: A Tree-Based Adaptive Solver for the Incompressible Euler Equations in Complex Geometries. In *Journal of Computational Physics*, **190**(2) (2003) 572–600.
16. Rosenfeld, A., Pfalz, J.L.: Distance Functions on Digital Pictures. In *Pattern Recognition*, **1** (1968) 33–61.
17. Saito, T., Toriwaki, J.: New Algorithms for n-dimensional Euclidean Distance Transformation. In *Pattern Recognition*, **27**(11) (1994) 1551–1565.
18. Samet, H.: A Quadtree Medial Axis Transform. In *Communications of the ACM*, **26**(9) (1983) 680–693.
19. Sintorn, I.M., Borgefors, G.: Weighted Distance Transforms for Volume Images Digitized in Elongated Voxel Grids. In *Pattern Recognition Letters*, **25**(5) (2004) 571–580.
20. Vacavant, A., Coeurjolly, D., Tougne, L.: Distance Transformation on Two-Dimensional Irregular Isothetic Grids. In *14th International Conference on Discrete Geometry for Computer Imagery (DGCI'2008)*, (2008) 238–249.
21. Vörös, J.: Low-Cost Implementation of Distance Maps for Path Planning Using Matrix Quadtrees and Octrees. In *Robotics and Computer-Integrated Manufacturing*, **17**(6) (2001) 447–459.

A Separability of the $\mathbb{I}$-BDT

In this section, we focus our interest in prooving the Proposition 1 of this report. It means that we have to make the relation between this separable process and the $\mathbb{I}$-BDT computation given in Equation 3.

Proposition 1 (Separable $\mathbb{I}$-BDT) *Let* $\mathbf{A}$ *be the associated irregular matrix of the 2-D* $\mathbb{I}$*-grid* $\mathbb{I}$*. Then the* $\mathbb{I}$*-BDT of* $\mathbb{I}$ *can be decomposed in two separable processes, and consists in computing the matrix* $\mathbf{B}$ *and* $\mathbf{C}$ *as follows:*

$$\mathbf{B}(i,j) = \min_x \big\{ \min\big(|T_X(i) - T_X(x) - H_R(x,j)|, |T_X(x) - T_X(i) - H_L(x,j)|\big) \; ; \; x \in \{0,...,n_1-1\},\ \mathbf{A}(x,j) = 0\big\},$$

$$\mathbf{C}(i,j) = \min_y \big\{\mathcal{G}_y(j);\ y \in \{0,\ldots,n_2-1\}\big\},$$

where $\mathcal{G}_y(j)$ *is the flattened parabola given by:*

$$\mathcal{G}_y(j) = \begin{cases} \mathbf{B}(i,y)^2 + (T_Y(j) - T_Y(y) - H_T(i,y))^2 & if\ T_Y(j) - T_Y(y) > H_T(i,y) \\ \mathbf{B}(i,y)^2 + (T_Y(y) - T_Y(j) - H_B(i,y))^2 & if\ T_Y(y) - T_Y(j) > H_B(i,y) \\ \mathbf{B}(i,y)^2 & else. \end{cases}$$

Proof. We first recall that Equation 3 considers the set of segments $\mathcal{S}$ belonging to the background/foreground boundary. As we consider $\mathbb{I}$-grids, $\mathcal{S}$ contains only vertical and horizontal segments. There is no points (or segments with a null-length) in it because we suppose the e-adjacency between cells [2]. We denote $\mathcal{S}_V$ the set of vertical segments and $\mathcal{S}_H$ horizontal ones. We now propose to write Equation 3 given by

$$\mathbb{I}\text{−BDT}(R) = \min_s \big\{d_e^2(p,s);\ s \in \mathcal{S}\big\}, \tag{3}$$

thanks to the sets $\mathcal{S}_V$ and $\mathcal{S}_H$:

$$\mathbb{I}\text{−BDT}(R) = \min\Big\{ \min_{s_h}\big\{d_e^2(p,s_h);\ s_h \in \mathcal{S}_H\big\}, \min_{s_v}\big\{d_e^2(p,s_v);\ s_v \in \mathcal{S}_V\big\}\Big\}. \tag{4}$$

Equation 4 also means that the shortest distance to the background/foreground boundary can be obtained by independently computing the distance along X to the nearest segment of $\mathcal{S}_V$ and the one along Y to the nearest segment of $\mathcal{S}_H$. Hence, we now consider the two steps of Equation 4:

1. Compute $\min_{s_v}\big\{d_e^2(p,s_v);\ s_v \in \mathcal{S}_V\big\}$ (5)
2. Compute $\min\Big\{ \min_{s_h}\big\{d_e^2(p,s_h);\ s_h \in \mathcal{S}_H\big\}, \min_{s_v}\big\{d_e^2(p,s_v);\ s_v \in \mathcal{S}_V\big\}\Big\}$. (6)

and draw its relation with Proposition 1, thanks to the irregular matrix. This data structure indeed allows us to treat an $\mathbb{I}$-grid along the two axis (see for example Figure 4 of this report). During the treatments we propose thereafter, we also do not want to increase the size of the matrix, to keep interesting space and time complexities in our proposed algorithm. We first consider the computation of the shortest distance with the vertical segments.

A.1 First Step along X Axis

Let $\mathbb{I}$ be a labeled 2-D $\mathbb{I}$-grid, and $\mathbf{A}$ its associated irregular matrix of size $n_1 \times n_2$. We consider in this section the nodes belonging to any row $j \in \{0, n_2 - 1\}$ of $\mathbf{A}$. This first step consists in finding the nearest segment $s_v \in \mathcal{S}_V$ of any node (i, j) of $\mathbf{A}$. As we work in one dimension, we thus have to find the *nearest points* of (i, j) belonging to the background/foreground boundary. In the irregular matrix, we add *virtual border nodes* to represent these points (see Figure 8). We now discuss two main cases:

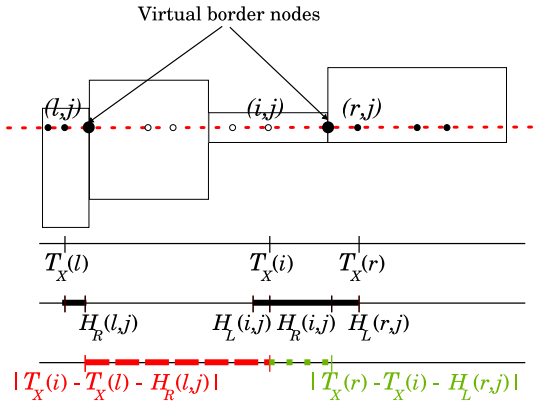


Fig. 8. For any node in a row j in $\mathbf{A}$, we consider the nearest point belonging to the background/foreground boundary. In this case, this point is obtained with the background node (r, j)

– The nearest virtual border node (v, j) does not coincide with any extra node of $\mathbf{A}$. In this case, we consider the nearest background node on its left (l, j) if (v, j) belongs to a right background cell border, or the nearest background node on its right (r, j) if (v, j) belongs to a left background cell border. As illustrated in Figure 8, these nodes are respectively on the left and on the right of the (i, j) node. Suppose now that the nearest node to (i, j) is (l, j). In this case, the attribute $H_R(l, j)$ represents the distance between (l, j) and the right border of the cell containing it. The distance between (v, j) and (i, j) is thus obtained with $T_X(i) - T_X(l) - H_R(l, j)$. In a similar way, if (r, j) is the nearest node to (i, j), we obtain the distance between (i, j) and the background/foreground boundary with $T_X(r) - T_X(i) - H_L(r, j)$ (see Figure 8). Hence, a general way to compute the shortest distance between (i, j) and all the virtual border nodes is to compute the minimal distance $d(x, i)$:

$$d(x, i) = \min\left(|T_X(i) - T_X(x) - H_R(x, j)|, |T_X(x) - T_X(i) - H_L(x, j)|\right)$$

for any background node (x, j), *i.e.* $\mathbf{A}(x, j) = 0$. If we suppose that (l, j) is the nearest background node, we thus deduce that:

$$\begin{aligned} & d(x,i) \geq \min\left(|T_X(i) - T_X(l) - H_R(l,j)|, |T_X(l) - T_X(i) - H_L(l,j)|\right) \\ \Leftrightarrow\ & d(x,i) \geq T_X(i) - T_X(l) - H_R(l,j),\ \forall x \in \{0, n_1 - 1\}, \mathbf{A}(x,j) = 0. \end{aligned}$$

If (r, j) is the nearest background node, we obtain:

$$\begin{aligned} & d(x,i) \geq \min\left(|T_X(i) - T_X(r) - H_R(r,j)|, |T_X(r) - T_X(i) - H_L(r,j)|\right) \\ \Leftrightarrow\ & d(x,i) \geq T_X(r) - T_X(i) - H_L(r,j),\ \forall x \in \{0, n_1 - 1\}, \mathbf{A}(x,j) = 0. \end{aligned}$$

– The nearest virtual border node (v, j) coincides with an existing extra node in $\mathbf{A}$. Since this node belongs to a vertical cell border, we have $H_R(v, j) = H_L(v, j) = 0$. With the notations given before, if (v, j) is on a right cell border, the distance between (v, j) and (i, j) is thus

$$T_X(i) - T_X(v) - H_R(v,j) = T_X(i) - T_X(l) - H_R(l,j) = T_X(i) - T_X(l).$$

In a similar way, if (v, j) is located on a left cell border, we obtain

$$T_X(v) - T_X(i) - H_L(v,j) = T_X(r) - T_X(i) - H_L(r,j) = T_X(r) - T_X(i).$$

As a consequence, we have proposed in this section a formulation of the first step of Equation 5 with the irregular matrix nodes. If we store the result of this process in a new matrix $\mathbf{B}$, we have shown that the computation of this distance:

$$\min_{s_v} \left\{ d_e^2(p, s_v);\ s_v \in \mathcal{S}_V \right\}$$

for any cell center p can be obtained by:

$$\begin{aligned} \mathbf{B}(i,j) = \min_x \big\{ & \min\left(|T_X(i) - T_X(x) - H_R(x,j)|, |T_X(x) - T_X(i) - H_L(x,j)|\right);\\ & x \in \{0, ..., n_1 - 1\},\ \mathbf{A}(x,j) = 0 \big\}, \end{aligned}$$

for any node $\mathbf{A}(i, j)$ of the irregular matrix. In practice, we do not have to add virtual nodes in the implementation of the irregular matrix. These supplemental nodes would have unnecessarily increase the size of the matrix. We now focus our interest on the second phase given in Equation 6, along the Y axis, where we compute the shortest distance with horizontal segments.

A.2 Second Step along Y Axis

We obtained in the previous phase the irregular matrix $\mathbf{B}$, storing the shortest distance between any node (i, j) and vertical segments of the background/foreground boundary. We now consider the nodes belonging to any column $i \in \{0, n_1 - 1\}$ of $\mathbf{B}$. As in the regular case [17], we can integrate the computation of the lower enveloppe of a set of parabolas to propagate the distance values along Y axis. In [20], we proposed to adapt this principle on an

irregular matrix to compute the $\mathbb{I}$-CDT. As in the previous phase, we consider virtual border nodes in column i. The distance to horizontal segments could thus be represented by classical parabolas on the existing nodes of the irregular matrix, and on these new nodes. In this case, Equation 6 could be written as in [20]:

$$\mathbf{C}(i,j) = \min_{y} \left\{ \mathbf{B}^*(i,y)^2 + (T_Y(j) - T_Y(y))^2;\ y \in \{0, ..., n_2 - 1\} \right\}. \quad (7)$$

In this equation, the term $\mathbf{B}^*(i,y)^2 + (T_Y(j) - T_Y(y))^2$ represents a classical parabola, and $\mathbf{B}^*$ is the irregular matrix with virtual border nodes. Let us now compare the representations of a background cell R

- in $\mathbf{B}^*$ with classical parabolas on each (normal, extra or virtual border) node (i,y) defined by

$$\mathcal{F}_y(j) = \mathbf{B}^*(i,y)^2 + (T_Y(j) - T_Y(y))^2,$$

- and in $\mathbf{B}$ with flattened parabolas on each (normal or extra) node (i,y), defined by

$$\mathcal{G}_y(j) = \begin{cases} \mathbf{B}(i,y)^2 + (T_Y(j) - T_Y(y) - H_T(i,y))^2 & if\ T_Y(j) - T_Y(y) > H_T(i,y) \\ \mathbf{B}(i,y)^2 + (T_Y(y) - T_Y(j) - H_B(i,y))^2 & if\ T_Y(y) - T_Y(j) > H_B(i,y) \\ \mathbf{B}(i,y)^2 & else. \end{cases}$$

In $\mathbf{B}^*$, as we consider a background cell, all the n nodes indexed $(i,y_1), (i,y_2), \ldots, (i,y_n)$ belonging to R have the same value $\mathbf{B}^*(i,y_k)^2 = 0,\ k \in \{1,n\}$. Hence, parabolas represented on each node (see Figure 9-a) are aligned along Z axis. Let us now consider a point of a parabola $\mathcal{F}_{y_k}(j),\ k \in \{1,n\}$, thus defined by $(T_Y(j), (T_Y(j) - T_Y(y_k))^2$. If $k = n$, then this point is lower bounded by the flattened parabola $\mathcal{G}_{y_{n-1}}(j)$ associated with the node $y_{k-1} = y_{n-1}$ in $\mathbf{B}$, since we have:

- If $T_Y(j) - T_Y(y_{n-1}) > H_T(i,y_{n-1})$, then

$$\begin{aligned} & \mathcal{G}_{y_{n-1}}(j) = (T_Y(j) - T_Y(y_{n-1}) - H_T(i,y_{n-1}))^2 \\ \Leftrightarrow\ & \mathcal{G}_{y_{n-1}}(j) = (T_Y(j) - (T_Y(y_{n-1}) + H_T(i,y_{n-1})))^2 \\ \Leftrightarrow\ & \mathcal{G}_{y_{n-1}}(j) = (T_Y(j) - T_Y(y_n))^2 = \mathcal{F}_{y_n}(j), \end{aligned}$$

  because, by construction of the nodes border attributes, we have $T_Y(y_{n-1}) + H_T(i,y_{n-1}) = T_Y(y_n)$ (see Figure 9-b).
- If $H_T(i,y_{n-1}) \leq T_Y(y_{n-1}) - T_Y(j) \leq H_B(i,y_{n-1})$, then

$$\mathcal{G}_{y_{n-1}}(j) = 0 \leq (T_Y(j) - T_Y(y_n))^2 = \mathcal{F}_{y_n}(j).$$

- If $T_Y(y_{n-1}) - T_Y(j) > H_B(i,y_{n-1})$, then

$$\begin{aligned} T_Y(j) <\ & T_Y(y_{n-1}) - H_B(i,y_{n-1}) && \leq T_Y(y_{n-1}) + H_T(i,y_{n-1}) = T_Y(y_n) \\ \Leftrightarrow 0 \leq\ & T_Y(y_{n-1}) - H_B(i,y_{n-1}) - T_Y(j) && \leq T_Y(y_n) - T_Y(j) \\ \Leftrightarrow 0 \leq\ & (T_Y(y_{n-1}) - H_B(i,y_{n-1}) - T_Y(j))^2 && \leq (T_Y(y_n) - T_Y(j))^2 \\ \Leftrightarrow 0 \leq\ & \mathcal{G}_{y_{n-1}}(j) && \leq \mathcal{F}_{y_n}(j). \end{aligned}$$

We can make similar reasonings if we study the first parabola in $\mathbf{B}^*$ associated with the node y_1 (*i.e.* $k = 1$). In this case, we may show that the flattened parabola $\mathcal{G}_{y_2}(j)$ in $\mathbf{B}$ lower bounds any points from the parabola $\mathcal{F}_{y_1}(j)$ in $\mathbf{B}^*$. If we consider a parabola $\mathcal{F}_{y_k}(j)$, $k \in \{2, n-1\}$, we can also simply show that for any Y-coordinate j, we have $\mathcal{G}_{y_k}(j) \leq \mathcal{F}_{y_k}(j)$. Our model thus permits to represent the computation of the $\mathbb{I}$-BDT distance beyond the borders of a background cell. Indeed, imagine an infinity of background nodes between the virtual border nodes in Figure 9-a. Hence, we should construct an infinity of classical parabolas in $\mathbf{B}^*$. Here, we choose to build flattened parabolas in $\mathbf{B}$, that are sufficient to bound them, and to correctly compute the $\mathbb{I}$-BDT.

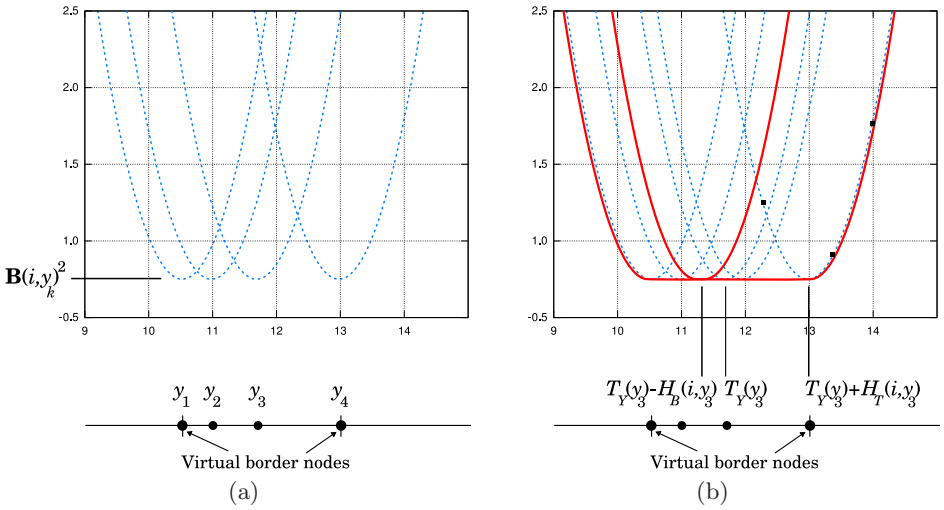


Fig. 9. Instead of adding virtual border nodes and represent classical parabolas (a), we use existing nodes and modelize flattened parabolas (b). We depict in (b) the values of the attributes of a background node, and its link with its associated flattened parabola. Points represented by squares belongs to the same parabola (associated with the virtual border node y_4), and are lower bounded by the same flattened parabola (associated with the node y_3). At the bottom of these plots, background nodes are represented by black circles at the corresponding Y-coordinate. Cell borders are also represented with vertical dashes

The justification of the model of flattened parabolas we have given for a background cell can also be applied for any set of $n \geq 1$ adjacent nodes indexed $(i, y_1), (i, y_2), \ldots, (i, y_n)$, where $\mathbf{B}(i, y_1) = \mathbf{B}(i, y_2) = \cdots = \mathbf{B}(i, y_n) \geq 0$. In a more general point of view, using flattened parabolas for each node of $\mathbf{B}$, instead of classical parabolas, permits to compute the $\mathbb{I}$-BDT, since they always lower bound classical parabolas. Furthermore, in the next chapter, we proove that

the intersection between two flattened parabolas can be determined (a single intersection point or an infinity), which proove the correctness of our model.

A.3 Conclusion

We have thus proved that Equation 3 can be considered as a separable minimization process, based on the irregular matrix structure. In Proposition 1, the computation of matrix $\mathbf{B}$ consists in a minimization process along X axis, while computing matrix $\mathbf{C}$ implies that we study the lower enveloppe of a set of flattened parabolas along Y axis. The $\mathbb{I}$-BDT definition can finally be compared as a generalization of the E^2DT on $\mathbb{I}$-grids, to get a cell border based distance map.

□

B Correctness of the $\mathbb{I}$-BDT Separable Computation

This section aims to proove the Lemma 1 given in Section 3 of this report, and a simple procedure to compute a valid intersection point between two flattened parabolas. In the rest of this section, we name *left branch* of a classical parabola $\mathcal{F}_\alpha$ the set of points $(y, \mathcal{F}_\alpha(y))$ such that $y \leq \alpha$. In a similar way, the *right branch* is the set of points $(y, \mathcal{F}_\alpha(y))$ such that $y \geq \alpha$. We need the following property of parabola branches:

Proposition 3 (Test between two branches of a parabola). *Let $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$ be two parabolas, with $\beta \neq \alpha$. If there exists an intersection point between $\mathcal{F}_\alpha$ and the left (respectively right) branch of $\mathcal{F}_\beta$, then there can not exist any intersection between $\mathcal{F}_\alpha$ and the right (left) branch of $\mathcal{F}_\beta$.*

We now recall the lemma to be proved:

Lemma 1 (Intersection between two flattened parabolas) *Let $\mathcal{G}_u$ and $\mathcal{G}_v$ be two flattened parabolas given by:*

$$\begin{aligned} &\mathcal{G}_u : \mathbb{R} \to \mathbb{R} \\ &\qquad y \to \mathcal{G}_u(y) = \begin{cases} g_u^2 + (u - y - lu_1)^2 & if\ u - y > lu_1 \\ g_u^2 + (y - u - lu_2)^2 & if\ y - u > lu_2 \\ g_v^2 & else \end{cases} \end{aligned}$$

and

$$\begin{aligned} &\mathcal{G}_v : \mathbb{R} \to \mathbb{R} \\ &\qquad y \to \mathcal{G}_v(y) = \begin{cases} g_v^2 + (v - y - lv_1)^2 & if\ v - y > lv_1 \\ g_v^2 + (y - v - lv_2)^2 & if\ y - v > lv_2 \\ g_v^2 & else \end{cases} \end{aligned}$$

where:

- *g_u, g_v, lu_1, lu_2, lv_1, lv_2, u and v are positive or null real numbers;*
- *if $u \geq v \geq 0$, then $u - lu_1 \geq v + lv_2$;*
- *if $0 \leq u \leq v$, then $u + lu_2 \leq v - lu_1$.*

The number of intersections between these parabolas is either one, either an infinity.

We can notice that a flattened parabola $\mathcal{G}_u$ is the composition of three distinct parts:

- A constant function with g_u^2 value at the centre, largely bounded by $u - lu_1$ and $u + lu_2$. We name it *base* of $\mathcal{G}_u$, and we denote it $\mathcal{B}_u$;
- the left branch of the parabola given by $g_u^2 + (y - u + lu_1)^2$ centred at the point $(g_u^2, u - lu_1)$. We denote the entire parabola $\mathcal{F}_{u-lu_1}(y) = \mathcal{F}_u^1(y)$, and $\widehat{\mathcal{F}}_u^1(y)$ is its left branch (its centre is excluded);

– the right branch of the parabola given by $(g_u^2 + (u - y - lu_2)^2$ centred at the point $(g_u^2, u + lu_2)$. We denote the entire parabola $\mathcal{F}_{u+lu_2}(y) = \mathcal{F}_u^2(y)$, and $\widehat{\mathcal{F}}_u^2(y)$ is its right branch (its centre is excluded).

We thus have $\mathcal{G}_u(y) = \mathcal{B}_u \cup \widehat{\mathcal{F}}_u^1(y) \cup \widehat{\mathcal{F}}_u^2(y)$, since the centres of the branches are excluded.

Proof. We consider thereafter that $g_u \geq g_v$ and that $v \geq u$ (similar reasonings may be done for others cases), and we study four different cases.

B.1 Case 1 (adjacency): $\mathbf{g_u = g_v}$, $\mathbf{u + lu_2 = v - lv_1}$, $\mathbf{lu_1 \neq 0, lu_2 \neq 0}$, $\mathbf{lv_1 \neq 0, lv_2 \neq 0}$

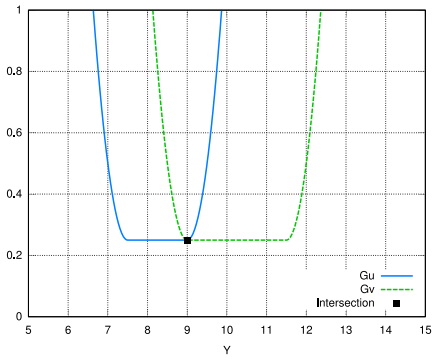


Fig. 10. Illustration of case 1 of our proof. The right part of $\mathcal{G}_u$, $\widehat{\mathcal{F}}_u^2$, can not intersect $\mathcal{G}_v$ because $lu_1, lu_2 \neq 0$

Here, the bases of the parabolas have the same value, and intersect at one extremity (since $u + lu_2 = v - lv_1$). By monotonicity, the right part of $\mathcal{G}_v$ can not intersect $\mathcal{G}_u$. Since the border attributes lv_1, lv_2 are not null, it can not cross neither $\widehat{\mathcal{F}}_u^1$ nor $\widehat{\mathcal{F}}_u^2$. The same is true of the left branch of $\mathcal{F}_u^1$ with $\mathcal{G}_v$. So, if there exist any intersection, it concerns either between the constant parts of the parabolas, or between $\widehat{\mathcal{F}}_u^2$ and $\widehat{\mathcal{F}}_v^1$. But the later configuration would imply that

$$\begin{aligned} & g_u^2 + (y - u - lu_2)^2 = g_v^2 + (v - y - lv_1)^2 \\ \Leftrightarrow\ & g_u^2 + (y - u - lu_2)^2 = g_u^2 + (-y + v - lv_1)^2 \\ \Leftrightarrow\ & (y - u - lu_2)^2 = (-y + v - lv_1)^2 \\ \Leftrightarrow\ & y = u + lu_2. \end{aligned}$$

they are positive values. This implieps that the intersection between the two branches is the point $(u + lu_2, g_u^2)$. This point is also the intersection between the bases of $\mathcal{G}_u$ and $\mathcal{G}_v$. As a consequence, in the case 1, the intersection between the flattened parabolas is a single point.

B.2 Case 2 (overlapping): $\mathbf{g_u = g_v}$, and either $\mathbf{v = u + lu_2}$, $\mathbf{lv_1 = lv_2 = 0}$, or $\mathbf{u = v - lv_1}$, $\mathbf{lu_1 = lu_2 = 0}$

We treat here a particular case of case 1, where one of the flattened parabola has no border attributes. We thus have for example $\mathcal{G}_v = \mathcal{F}_v^2 = \mathcal{F}_v^1 = \mathcal{F}_v$ if $v = u + lu_2$ et $lv_1 = lv_2 = 0$. We consider this specific case, and configuration where $\mathcal{G}_u = \mathcal{F}_u$ may be treated in a similar way. As $\mathcal{G}_v = \mathcal{F}_v$, and since $v = u + lu_2$, the right parabola of $\mathcal{G}_u$ coincides with $\mathcal{G}_v$, because

$$\begin{aligned} & \mathcal{F}_u^2(y) = g_u^2 + (y - u - lu_2)^2 \\ \Leftrightarrow\ & \mathcal{F}_u^2(y) = g_v^2 + (y - v)^2 \\ \Leftrightarrow\ & \mathcal{F}_u^2(y) = \mathcal{F}_v(y) = \mathcal{G}_v(y). \end{aligned}$$

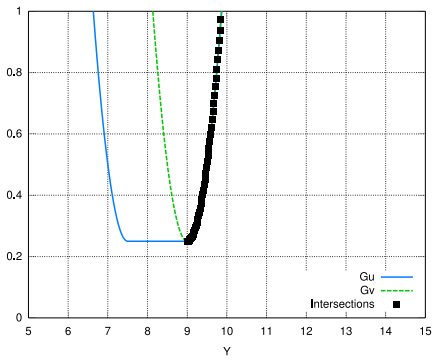


Fig. 11. Illustration of case 2 of our proof. As the right parabola of $\mathcal{G}_u$ coincides with $\mathcal{G}_v$, there is an infinity of intersections

This naturally implies that there exist an infinity of intersections between $\widehat{\mathcal{F}}_u^2(y)$, the right branch of $\mathcal{G}_u$, and $\mathcal{G}_v$. To get a lower bound of this set of points, we can not consider any intersection between $\mathcal{G}_v$ and $\widehat{\mathcal{F}}_u^1(y)$ (left branch) because, as $lu_1, lu_2 \neq 0$, $\mathcal{G}_v$ only intersects the right branch of $\mathcal{F}_u^1(y)$. By Proposition 3, we deduce that $\mathcal{G}_v$ does not cross the left branch of $\mathcal{F}_u^1(y)$, *i.e.* $\widehat{\mathcal{F}}_u^1(y)$. The lowest point thus belongs to $\mathcal{B}_u$, and has coordinates (v, g_u^2). We conclude the same if we suppose that $\mathcal{G}_u = \mathcal{F}_u$ ($lu_1 = lu_2 = 0$), and we obtain that $\mathcal{F}_v^1(y) = \mathcal{G}_u(y)$. We can also suppose that $u = v$ and $lu_1 = lu_2 = lv_1 = lv_2 = 0$. With these conditions, we consider $\mathcal{G}_u = \mathcal{F}_u$ and $\mathcal{G}_v = \mathcal{F}_v = \mathcal{F}_u$ as coincident classical parabolas. As indicated in Proposition 2, they share an infinity of points. As a consequence, flattened parabolas described in this case 2 have an infinity of intersection points, with a infimum.

B.3 Case 3 (general) : $\mathbf{g_u > g_v}$

As we now suppose that $v > u$, the right branch of $\mathcal{G}_v$ ($\widehat{\mathcal{F}}_v^2$) can not cross the parabola $\mathcal{G}_u$, by monotonicity (Proposition 2). Let now suppose for example that $\widehat{\mathcal{F}}_v^2$ intersects the right branch of $\mathcal{G}_u$ (*i.e.* $\widehat{\mathcal{F}}_u^2$). We have:

$$\begin{aligned} & g_u^2 + (y - u - lu_2)^2 = g_v^2 + (y - v - lv_2)^2 \\ \Leftrightarrow\ & g_u^2 - g_v^2 = (y - v - lv_2)^2 - (y - u - lu_2)^2, \end{aligned}$$

and since $g_u > g_v$, we obtain

$$\begin{aligned} & (y - v - lv_2)^2 - (y - u - lu_2)^2 > 0 \\ \Leftrightarrow\ & y - v - lv_2 > y - u - lu_2 \\ \Leftrightarrow\ & v + lv_2 \leq u + lu_2. \end{aligned}$$

But, in our definition of flattened parabola, we have $v \geq u \Rightarrow u + lu_2 \leq u - lv_1$, and, moreover, $u - lv_1 \leq v + lv_2$. Hence, we have both $u + lu_2 \leq v + lv_2$ and $v + lv_2 \leq u + lu_2$, which leads to a contradiction since $v > u$. We can show in a similar way that $\widehat{\mathcal{F}}_v^2$ cannot intersect the others parts of $\mathcal{G}_u$, *i.e.* its base $\mathcal{B}_u$ and its left branch $\widehat{\mathcal{F}}_u^1$. We also know that the base of $\mathcal{G}_v$ (given by $z = g_v^2$) cannot cross $\mathcal{G}_u$, because $g_u > g_v$. Indeed, the Y-coordinates of $\mathcal{G}_u$ points are all greater than $g_u^2 > g_v^2$.

As a consequence, we can now consider the intersection between the left branch of $\mathcal{G}_v$ (*i.e.* $\widehat{\mathcal{F}}_v^1(y)$) and the parabola $\mathcal{G}_u$. To determine where this branch cuts $\mathcal{G}_v$, we first compute the intersection between $\widehat{\mathcal{F}}_v^1(y)$ and the line given by $z = g_u^2$:

$$\begin{aligned} & g_v^2 + (v - y - lv_1)^2 = g_u^2 \\ \Leftrightarrow\ & (v - y - lv_1)^2 = g_u^2 - g_v^2 \\ \Leftrightarrow\ & v - y - lv_1 = \sqrt{g_u^2 - g_v^2} \\ \Leftrightarrow\ & y = v - lv_1 - \sqrt{g_u^2 - g_v^2}. \end{aligned}$$

The square root operation is possible sicne we treat positive values. Thereafter, we consider the different values of y that we denote $\bar{y}$ to distinguish it with the global variable y. We now study:

- the intersection between $\widehat{\mathcal{F}}_v^1(y)$ and the line given by $y = u - lu_1$;
- the intersection between $\widehat{\mathcal{F}}_v^1(y)$ and the line given by $y = u + lu_2$.

We can compute these crossings because we have $v - lv_1 \geq u + lu_2 \geq u - lu_1$. By monotonicity, we clearly have:

$$\begin{aligned} & 0 \leq \widehat{\mathcal{F}}_v^1(u + lu_2) \leq \widehat{\mathcal{F}}_v^1(u - lu_1) \\ \Leftrightarrow\ & 0 \leq \widehat{\mathcal{F}}_v^1(u + lu_2) - g_v^2 \leq \widehat{\mathcal{F}}_v^1(u - lu_1) - g_v^2, \end{aligned}$$

The last inequality is true, and is due to the construction of the left branch of $\mathcal{G}_v$. We denote:

$$\begin{aligned} F_1 &= \widehat{\mathcal{F}}_v^1(u - lu_1) - g_v^2, \\ F_2 &= \widehat{\mathcal{F}}_v^1(u + lu_2) - g_v^2, \text{ with} \\ 0 &\leq F_2 \leq F_1. \end{aligned}$$

As $g_u > g_v$, we now compare these two values with $g_u^2 - g_v^2 = G > 0$ (an illustration is depicted in Figure 12). For each case (in respect to the place of G in the previous equation), we show a representation fo these three elements, to make the reading easier. We also draw the relation with $\bar{y}$, and exactly determine the intersection between the concerned flattened parabolas.

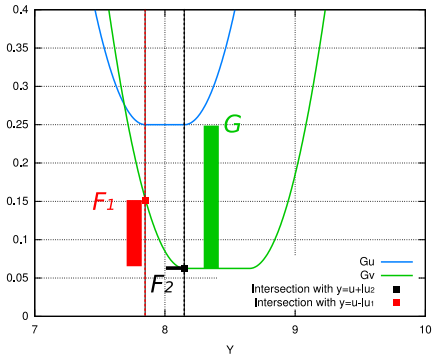


Fig. 12. Comparison between the values F_1, F_2 and G in our proof. We have here $g_u^2 - g_v^2 > \widehat{\mathcal{F}}_v^1(u - lu_1) - g_v^2 > \widehat{\mathcal{F}}_v^1(u + lu_2) - g_v^2 \Leftrightarrow G > F_1 > F_2 > 0$, with F_2 nearby 0. We can now determine the intersection between $\mathcal{G}_u$ and $\mathcal{G}_v$. The case presented in this figure (the left branch of $\mathcal{G}_v$ crosses the left branch of $\mathcal{G}_u$) is considered in the paragraph "Case 3b"

Case 3a: $\mathbf{F_2 \leq G \leq F_1}$ This double inequality implies that

$$\begin{aligned} \widehat{\mathcal{F}}_v^1(u + lu_2) - g_v^2 \leq \quad & g_u^2 - g_v^2 \quad \leq \widehat{\mathcal{F}}_v^1(u - lu_1) - g_v^2 \\ \Leftrightarrow (v - u - lu_2 - lv_1)^2 \leq \quad & g_u^2 - g_v^2 \quad \leq (v - u + lu_1 - lv_1)^2 \\ \Leftrightarrow v - u - lu_2 - lv_1 \leq \quad & \sqrt{g_u^2 - g_v^2} \quad \leq v - u + lu_1 - lv_1 \\ \Leftrightarrow v - u - lu_2 - lv_1 \leq \quad & \sqrt{g_u^2 - g_v^2} \quad \leq v - u + lu_1 - lv_1 \\ \Leftrightarrow -u - lu_2 \leq -v + lv_1 + \sqrt{g_u^2 - g_v^2} & \leq -u + lu_1 \\ \Leftrightarrow u - lu_1 \leq \quad & \bar{y} \quad \leq u + lu_2. \end{aligned}$$

Which clearly means that the parabola $\mathcal{G}_v$ intersects the base of $\mathcal{G}_u$, $\mathcal{B}_u$, and more exactly at the point $(\bar{y}, g_u^2)$. So, $\mathcal{G}_v$ intersects the right branch of $\mathcal{F}_u^1(y)$,

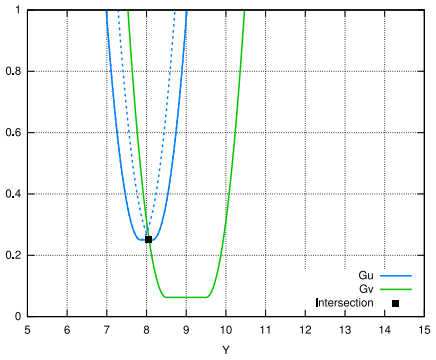


Fig. 13. Illustration of case 3a of our proof. $\mathcal{G}_v$ intersects the right branch of $\mathcal{F}_u^1(y)$ (left classical parabola classique in dotted lines). And it crosses the left branch of $\mathcal{F}_u^2(y)$ (right classical parabola classique in dotted lines)

because $u - lu_1 \leq \bar{y}$. By monotonicity, (and Proposition 3), it cannot reach the other branch of this parabola. In a similar way, $\mathcal{G}_v$ crosses the left branch of $\mathcal{F}_u^2(y)$, since $\bar{y} \leq u + lu_2$, and cannot cross its right branch. As a conclusion, $\mathcal{G}_v$ crosses at a single point the parabola $\mathcal{G}_u$ in $\mathcal{B}_u$, and any other intersection point exists between them. Those remarks can be enounced when $lu_1 = lu_2 = 0$. In this case, the intersection point is (u, g_u^2), and $\mathcal{G}_v$ cannot cross the branches of $\mathcal{G}_u$ (which exclude this point).

Case 3b : $\mathbf{F_2 \leq F_1 \leq G}$ We can enounce a double inequality as previously:

$$\begin{aligned} \widehat{\mathcal{F}}_v^1(u + lu_2) - g_v^2 \leq \widehat{\mathcal{F}}_v^1(u - lu_1) - g_v^2 \leq g_u^2 - g_v^2 \\ \Leftrightarrow \bar{y} \leq \quad u - lu_1 \quad \leq u + lu_2. \end{aligned}$$

The case where $\bar{y} = u - lu_1$ directly implies that the parabola $\mathcal{G}_v$ crosses the base of $\mathcal{G}_u$. One can refer to the previous case 3a to compute the intersection point. If $\bar{y} < u - lu_1$, by the construction of flattened parabolas, it is clear that the intersection between $\mathcal{G}_u$ and $\mathcal{G}_v$ only exists with the left branch of $\mathcal{G}_u$ (*i.e.* $\widehat{\mathcal{F}}_u^2(y)$). Indeed, as $\bar{y} < u - lu_1$, any crossing with the base $\mathcal{B}_u$ is impossible (it is a constant function defined for $u + lu_2 \geq y \geq u - lu_1$). Then, suppose that $\mathcal{G}_v$ (and more precisely its left branch $\widehat{\mathcal{F}}_v^1(y)$) crosses the right branch of $\mathcal{G}_u$ ($\widehat{\mathcal{F}}_u^1(y)$), we have:

$$\begin{aligned} \widehat{\mathcal{F}}_u^2(y) &= \widehat{\mathcal{F}}_v^1(y) \\ \Leftrightarrow g_u^2 + (u - y - lu_2)^2 &= g_v^2 + (v - y - lv_1)^2 \\ \Leftrightarrow g_u^2 - g_v^2 &= (v - y - lv_1)^2 - (u - y - lu_2)^2. \end{aligned}$$

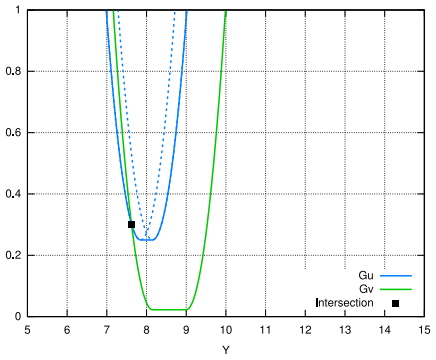


Fig. 14. Illustration of the case 3b in our proof. If $\bar{y} < u - lu_1$, it is clear that the intersection exists between $\mathcal{G}_v$ and the left branch of $\mathcal{G}_u$. Hence, $\mathcal{G}_v$ can not intersect any other part of $\mathcal{G}_u$

But we ave supposed that $F_2 \leq F_1 \leq G = g_u^2 - g_v^2$. We deduce that:

$$\begin{aligned} F_2 = \mathcal{F}_v^1(u - lu_2) - g_v^2 &\leq (v - y - lv_1)^2 - (u - y - lu_2)^2 \\ \Leftrightarrow (v - lv_1 - u - lu_2)^2 &\leq (-2y + v - lv_1 + u + lu_2)(v - lv_1 - u - lu_2) \\ \Leftrightarrow v - lv_1 - u - lu_2 &\leq -2y + v - lv_1 + u + lu_2 \\ \Leftrightarrow y &\leq u + lu_2, \end{aligned}$$

which contradicts our hypothesis. Its is impossible that the intersection point belongs to the right branch of $\mathcal{G}_u$, defined on $y > u + lu_2$, whereas we have $y \leq u + lu_2$. Those remarks are valid if $lu_1 = lu_2 = 0$ (in this case, $\mathcal{G}_u$ is a classical parabola, and we can also use the proposition 3). We can conclude that there exists only one intersection point between the two parabolas $\mathcal{G}_u$ and $\mathcal{G}_v$, and its coordinates are $(\bar{y}, \widehat{\mathcal{F}}_u^1(\bar{y}))$.

Case 3c : $\mathbf{G \leq F_2 \leq F_1}$ As the previous case, we bound $\bar{y}$ thanks to the following double inequality:

$$\begin{aligned} g_u^2 - g_v^2 \leq \widehat{\mathcal{F}}_v^1(u + lu_2) - g_v^2 &\leq \widehat{\mathcal{F}}_v^1(u - lu_1) - g_v^2 \\ \Leftrightarrow u - lu_1 \leq \qquad u + lu_2 \qquad &\leq \bar{y}. \end{aligned}$$

When $y = u + lu_2$, we can apply the proof of the case 3a to compute the intersection point between the parabola $\mathcal{G}_v$ and the base of $\mathcal{G}_u$, *i.e.* $\mathcal{B}_u$. If we now suppose that $u > u + lu_2$, we can notice that the intersection with $\mathcal{B}_u$ is impossible, because this constant function is defined for $u - lu_1 \leq y \leq u + lu_2$. Suppose now that the left branch of $\mathcal{G}_v$ (*i.e.* $\widehat{\mathcal{F}}_v^1(y)$) crosses the left branch of $\mathcal{G}_u$

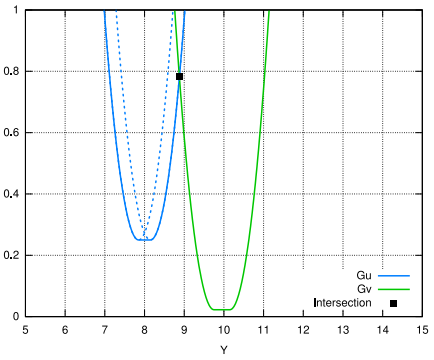


Fig. 15. Illustration of the case 3c in our proof. If $\bar{y} > u + lu_2$, it is clear that the intersection exist between $\mathcal{G}_v$ and the right branch of $\mathcal{G}_u$. Hence, $\mathcal{G}_v$ can not intersect any other part of $\mathcal{G}_u$

(*i.e.* $\widehat{\mathcal{F}}_u^1(y)$). We have:

$$\begin{aligned} \widehat{\mathcal{F}}_u^1(y) &= \widehat{\mathcal{F}}_v^1(y) \\ \Leftrightarrow g_u^2 + (u - y - lu_1)^2 &= g_v^2 + (v - y - lv_1)^2 \\ \Leftrightarrow g_u^2 - g_v^2 &= (v - y - lv_1)^2 - (u - y - lu_1)^2. \end{aligned}$$

Since we have $G \leq F_2 \leq F_1$, we deduce that:

$$\begin{aligned} (v - y - lv_1)^2 - (u - y - lu_1)^2 \leq F_1 = \mathcal{F}_v^1(u - lu_1) - g_v^2 \\ \Leftrightarrow (v - lv_1 - u + lu_1)(-2y + v - lv_1 + u - lu_1) \leq (v - lv_1 - u + lu_1)^2 \\ \Leftrightarrow -2y + v - lv_1 + u - lu_1 \leq v - lv_1 - u + lu_1 \\ \Leftrightarrow y \geq u - lu_1, \end{aligned}$$

which contradicts our hypothesis (an intersection exists on the left branch of $\mathcal{G}_u$, defined for $y < u - lu_1$), and we obtain that $y \geq u + lu_1$. As a consequence, there exists only one intersection between $\mathcal{G}_v$ and $\mathcal{G}_v$, and is located at $(\bar{y}, \widehat{\mathcal{F}}_u^2(\bar{y})$. Moreover, we obtain the same result if $lu_1 = lu_2 = 0$ (as in the case 3b).

B.4 Case of non-intersection between two flattened parabolas

In all the cases we treated up to now, there is a last configuration, where $u = v$, $lu_1 = lu_2 = lv_1 = lv_2 = 0$ (which implies that these are two classical parabolas $\mathcal{F}_u$ and $\mathcal{F}_v$ aligned along the Y axis), with $g_u > g_v$. This case is similar with case 2, where g_u, g_v are strictly different values. Suppose that there exist an intersection point between them, we have:

$$\begin{aligned} \mathcal{F}_u &= \mathcal{F}_v \\ \Leftrightarrow g_u^2 + (y - u)^2 &= g_v^2 + (y - v)^2 \\ \Leftrightarrow 0 < g_u^2 - g_v^2 &= (y - v)^2 - (y - u)^2 = 0, \end{aligned}$$

and we lead to a contradiction since $g_u > g_v$ and $u = v$. We have depicted the only case of empty intersection between two flattened parabolas.

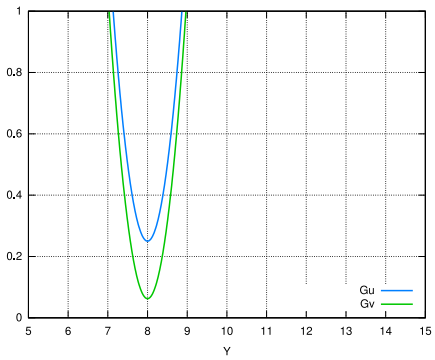


Fig. 16. Illustration of non-intersection case between two flattened parabolas. If $u = v$, $lu_1 = lu_2 = lv_1 = lv_2 = 0$, and $g_u > g_v$, then there is no intersection between them

B.5 Conclusion

Thanks to the different configurations we have treated (Cases 1, 2, 3 and 4), we have proved that two flattened parabolas can share either one, or an infinity of points. This property permits to directly conclude that Lemma 2 of this report is correct. More precisely, the minimization process along Y axis computes the lower enveloppe of a set of flattened parabolas, giving a correct distance map based on cells borders.

As a conclusion, we now propose to describe a routine to compute a valid intersection point between them, *i.e.* a point that then allows us to correctly order them:

i) Compute the intersection point, with X-coordinate $\bar{y}$, between $\mathcal{G}_v$ and the line given by $z = g_u^2$;
ii) If $u - lu_1 \leq \bar{y} \leq u + lu_2$, return the point $(\bar{y}, g_u^2)$;
iii) If $\bar{y} \leq u - lu_1 \leq u + lu_2$, return the intersection point between $\widehat{\mathcal{F}}_v^1$ and $\widehat{\mathcal{F}}_u^1$;
iv) If $u - lu_1 \leq u + lu_2 \leq \bar{y}$, return the intersection point between $\widehat{\mathcal{F}}_v^1$ and $\widehat{\mathcal{F}}_u^2$.

The $\bar{y}$ point introduced in step i) comes from Case 3 of the previous proof. As in the regular case (and the use of classical parabolas), it is thus possible to order a set of flattened parabolas, by considering their intersection points. This routine is usefull to implement the adaptation of T. Saito *et al.*'s E^2DT algorithm [17] to compute the $\mathbb{I}$-BDT.

□