A combined multi-scale/irregular algorithm for the vectorization of noisy digital contours

Antoine Vacavant[a], Tristan Roussillon[b,c], Bertrand Kerautret[d,e], Jacques-Olivier Lachaud[e]

[a] *Clermont Université, Université d'Auvergne, ISIT, F-63001, France*
[b] *Université de Lyon, CNRS*
[c] *Université Lyon 2, LIRIS, UMR5205 CNRS, F-69676, France*
[d] *Université de Nancy, LORIA, UMR7503 CNRS, F-54506, France*
[e] *Université de Savoie, LAMA, UMR5127 CNRS, F-73376, France*

Abstract

This paper proposes and evaluates a new method for reconstructing a polygonal representation from arbitrary digital contours that are possibly damaged or coming from the segmentation of noisy data. The method consists in two stages. In the first stage, a multi-scale analysis of the contour is conducted so as to identify noisy or damaged parts of the contour as well as the intensity of the perturbation. All the identified scales are then merged so that the input data is covered by a set of pixels whose size is increased according to the local intensity of noise. The second stage consists in transforming this set of resized pixels into an irregular isothetic object composed of an ordered set of rectangular and axis-aligned cells. Its topology is stored as a Reeb graph, which allows an easy pruning of its unnecessary spurious edges. Every remaining connected part has the topology of a circle and a polygonal representation is independently computed for each of them. Four different geometrical algorithms, including a new one, are reviewed for the latter task. These vectorization algorithms are experimentally evaluated and the whole method is also compared to previous works on both synthetic and true digital images. For fair comparisons, when possible, several error measures between the reconstruction and the ground truth are given for the different techniques.

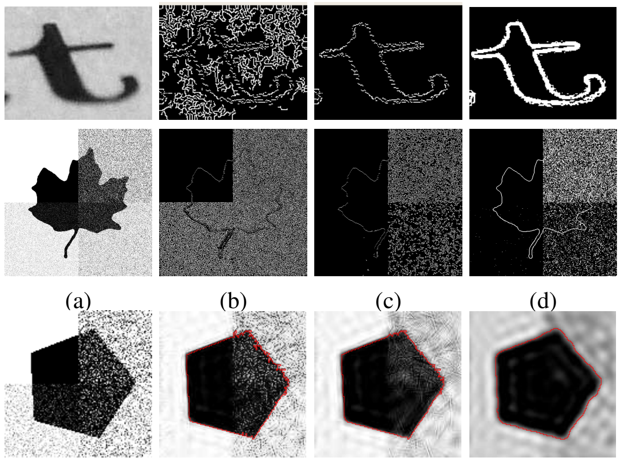
Keywords:
Noisy object analysis, Multi-scale noise detection, Irregular grids, Reeb graph

1. Introduction

The vectorization (*i.e.* reconstruction into line segments) of digital objects obtained from segmentation, digitization or scanning processes is a very common task in many image analysis systems such as optical character recognition (OCR), license plate recognition (LPR), sketch recognition, *etc.* [1, 8, 15, 38, 33, 39, 40]. The development of raster-to-vector (R2V) algorithms is in constant progress, responding to both technical and theoretical challenges [32]. Indeed, in real-life applications, digital objects are not perfect digitizations of ideal shapes but present noise, disconnections, irregularities, *etc.*

To process this kind of image data, additional information is provided such as *a priori* knowledge on studied shapes (for instance, shapes are letters in OCR) or user supervision. For low level image processing, classic approaches of contour (or edge) detection generally need an external parameter that has to be tediously tuned, and the output has to be filtered and post-processed [3, 10] (see Figure 1 for an example with the Canny edge detector and the Sobel operator with also a recent algorithm of edge noise removal [14]).

The noisy digital contour (or a thick digital curve around it) can be partitioned into thick (or blurred) segments [11, 12]. Such approaches require a global thickness parameter and thus cannot handle contours along which the amount of perturbation or noise is not uniform (e.g. see Figure 1(a), top and bottom). The document vectorization method of [15] also assumes rather uniform noise so that filtering and skeletonization are enough to take care of it. Other methods like [19, 29], which are based



(a) (b) (c) (d)

(e) (f) $\sigma = 20$ (g) $\sigma = 30$ (h) $\sigma = 40$

Figure 1: The Canny edge detector (b-c) with two parameters and the Sobel operator (d) applied on two images. For the first image, even if we could obtain an interesting result, a post-process is necessary to filter the output of the detectors in order to compute a linear contour. The second very noisy image cannot be efficiently handled by these techniques used alone, even with various parameters. The third row shows a recent edge noise removal approach applied with several parameters [14].

on different principles, also require a global scale parameter to compute polygonal reconstructions. This parameter is again related to the amount of noise in the image.

We proposed in a previous work [37] a novel *unsupervised technique*, divided into two main stages. We first used the pixel resizing algorithm based on the multi-scale noise detector introduced in [17]. This set of resized pixels is transformed into an irregular isothetic object composed of rectangular and axis-aligned cells. The topology is stored into a Reeb graph [25]. The object is then analyzed and vectorized using two geometrical algorithms, both based on the preimage of straight parts (*i.e.* sequence of cells that can be passed through by a straight line). These two polygonalization algorithms are an improvement of the visibility cone approach of [34].

Our system is comparable with the work of [26], where is introduced a polygonalization technique based on a pixel resizing step, combined with a generalized preimage algorithm. However, this approach mixes up noise, arithmetic artefacts and high curvature features when trying to detect noisy parts of contours. It also needs a very complex topological control process [27], represented as a skeleton, to handle objects not homotopic to a cycle.

In this paper, we extend the approach introduced in [37], along three directions. First, the Reeb graph, which contains the topology of the irregular object, is better exploited in order to get a polygonal representation of the input digital contour that is homeomorphic to a circle (one connected component and one hole) and such that exactly two edges are incident to each vertex. This filtering step also informs us if the processed irregular object can be interpreted as a single cycle, and may loop back to the multi-scale noise detector to have an analysis at a finer scale. Then, we propose another geometrical algorithm that minimizes, for each k-arc (*i.e.* parts of connected cells), the length of the polygonal representation. The output of this algorithm turns out to be a good trade-off between minimizing the number of vertices and minimizing the reconstruction error. Finally, we conduct a larger amount of quantitative comparisons with other vectorization techniques in order to validate our approach. We illustrate the global processing chain of our system in Figure 2.

After recalling basic definitions about irregular isothetic objects and their construction from a noisy digital contour (Section 2), we show in Section 3 how to filter the obtained irregular object using its Reeb graph in order to get a faithful representation of the input digital contour. In Section 4, the vectorization techniques of [34, 37] is recalled and we introduce a novel approach based on the minimal-length polygon inscribed in a polygonal object. As an experimental validation, we compare the different reconstruction algorithms and compare the whole method to other vectorization techniques in Section 5. We also propose a hybrid polygonalization method that combines two formerly presented polygonalization techniques: it exploits the flat part or curved part tags that are a byproduct of the multi-scale analysis.

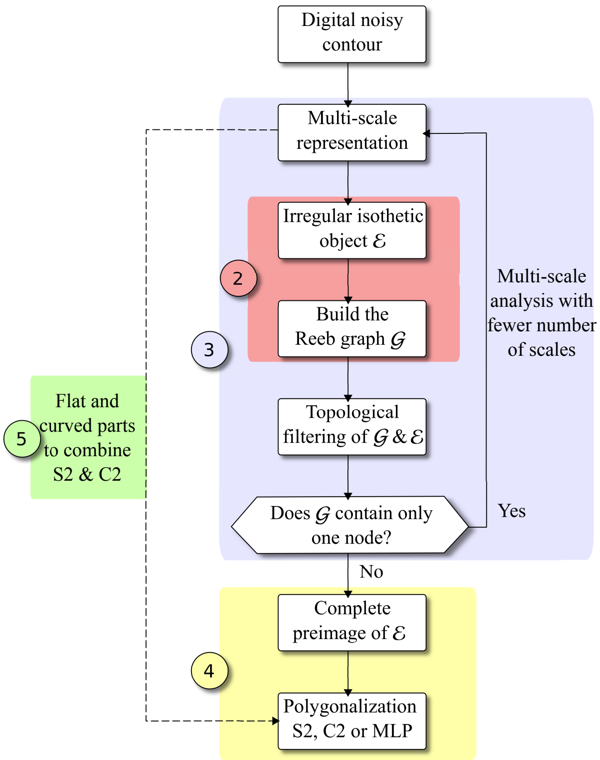


Figure 2: Global processing chain of our system. Each part is also labelled with the section number where it is described.

2. Preamble and previous work

2.1. Definitions

In this section, we first recall the concept of irregular isothetic grids ($\mathbb{I}$-grids) in 2-D, with the following definitions [7, 36].

Definition 1 (2-D $\mathbb{I}$-grid)**.** *Let $\mathcal{R}$ be a closed rectangular subset of $\mathbb{R}^2$. A 2-D $\mathbb{I}$-grid G is a tiling of $\mathcal{R}$ with closed rectangular cells whose edges are parallel to the X and Y axes, and whose interiors have a pairwise empty intersection. The position of each cell R is given by its center point $(x_R, y_R) \in \mathbb{R}^2$ and its length along X and Y axes by $(l_R^x, l_R^y) \in \mathbb{R}_+^{*\,2}$.*

Definition 2 (ve−adjacency and e−adjacency)**.** *Let R_1 and R_2 be two cells. R_1 and R_2 are ve−adjacent (vertex and edge adjacent) if :*

$$\text{or}\begin{cases} |x_{R_1}-x_{R_2}| = \frac{l^x_{R_1}+l^x_{R_2}}{2} \text{ and } |y_{R_1}-y_{R_2}| \le \frac{l^y_{R_1}+l^y_{R_2}}{2} \\ |y_{R_1}-y_{R_2}| = \frac{l^y_{R_1}+l^y_{R_2}}{2} \text{ and } |x_{R_1}-x_{R_2}| \le \frac{l^x_{R_1}+l^x_{R_2}}{2} \end{cases}$$

R_1 and R_2 are e−adjacent (edge adjacent) if we consider an exclusive "or" and strict inequalities in the above ve−adjacency definition. k may be interpreted as e or ve in the following definitions.

A k-path from R to R' is a sequence of cells $(R_i)_{1\le i\le n}$ with $R = R_1$ and $R' = R_n$ such that for any i, $2 \le i < n$, R_i is k-adjacent to R_{i-1} and R_{i+1}.

Definition 3 (k-arc)**.** *Let $A = (R_i)_{1\leq i\leq n}$ be a k-path from R_1 to R_n. Then A is a k-arc iff each cell R_i has exactly two k-adjacent cells in A except R_1 and R_n which have only one k-adjacent cell in A. The cells R_1 and R_n are called the* extremities *of A.*

Definition 4 (k-object)**.** *Let $\mathcal{E}$ be a set of cells, $\mathcal{E}$ is a k-object iff for each couple of cells $(R, R') \in \mathcal{E} \times \mathcal{E}$, there exists a k-path between R and R' in $\mathcal{E}$.*

We consider an order relation based on the cells borders. We denote the left, right, top and bottom borders of a cell R respectively R^L, R^R, R^T and R^B. The abscissa of R^L, for example, is equal to $x_R - (l^x_R/2)$. In the following, we also denote by $\leq_x$ (resp. $\leq_y$) the natural order relation along X (resp. Y) axis. It is legitimate to use the order $\leq_x$ on left and right borders of cells and the order $\leq_y$ on top and bottom borders of cells.

Definition 5 (Order relation on an $\mathbb{I}$-grid)**.** *Let R_1 and R_2 be two cells of an $\mathbb{I}$-grid G. We define the total order relation $\leq^L$, based on the cells borders:*

$$\forall R_1, R_2 \in G,$$
$$R_1 \leq^L R_2 \Leftrightarrow R_1^L <_x R_2^L \vee \left(R_1^L =_x R_2^L \wedge R_1^T \leq_y R_2^T\right).$$

This order relation is of great importance both for the Reeb graph computation and in the polygonalization stage, where it leads to linear-time geometrical algorithms.

2.2. Representation of the topology of an irregular object

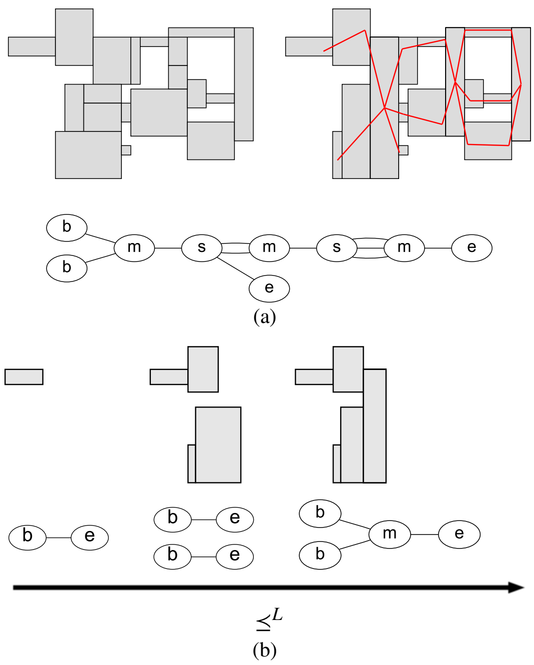


Figure 3: (a) An example of an irregular object $\mathcal{E}$ (left), the final recoded structure with arcs, the obtained polygonalization (right) and the Reeb graph associated to the order defined on $\mathcal{E}$ (bottom) [35]. In (b), we show the recognized k-arcs and the associated Reeb graph for some iterations of this algorithm, with respect to the $\leq^L$ order.

The procedure that transforms any irregular object into a graph of k-arcs is fully described in [34], and we recall here the main principles of this transformation.

The k-object $\mathcal{E}$ is scanned from left to right according to the order induced by $\leq^L$, given in Definition 5 (see Figure 3 for an example). The Reeb graph [25] of $\mathcal{E}$, which is a way of representing its topology, is built incrementally as follows. At the beginning, the intersection between $\mathcal{E}$ and the scanning vertical line has only one connected part and the Reeb graph is created with one arc between two nodes (b for begin and e for end). If a connected part splits into several parts, we add a node (s for split) from which start as many arcs as there are parts. Conversely, if two connected parts merge, we link the corresponding arcs to a node (m for merge) (see Figure 3).

Moreover, the initial set of cells is recoded into a new one (without changing the shape of the object however) so that each arc of the Reeb graph corresponds to a k-arc having cells of increasing left border. We merge with the cell having the smallest left border all its k-adjacent cells by using the following update procedure.

Update procedure. Let A be a k-arc in $\mathcal{E}$ such that R_1 is its rightmost extremity. Let R_2 be a cell of $\mathcal{E}$ that is k-adjacent to A at R_1 and such that $R_1^L <_x R_2^L$ (and thus should be added to A). If $R_2^L =_x R_1^R$, one just add R_2 to A after R_1, else the procedure *updates* the k-arc A at R_1, and may recode the end of A. For that, it first builds the *greatest common rectangle* (GCR) F_2 of R_1 and R_2. This GCR is the greatest rectangle that can be contained in $R_1 \cup R_2$ [34]:

Definition 6 (Greatest common rectangle)**.** *Let R_1 and R_2 be two k-adjacent rectangles. The rectangle F_2 is the greatest common rectangle (GCR) of R_1 and R_2 iff*

(i) $F_2 \subseteq R_1 \cup R_2$;

(ii) $R_1 \cap R_2 \subseteq F_2$;

(iii) there is no rectangle greater than F_2 by inclusion respecting i), ii).

Then the rectangles $R_1 - F_2$ and $R_2 - F_2$ are denoted by F_1 and F_3 respectively. The k-arc A is finally updated with respect to five main configurations, by replacing R_1 in A by the sequence (F_1, F_2, F_3) (see Figure 4, empty rectangles are not added).

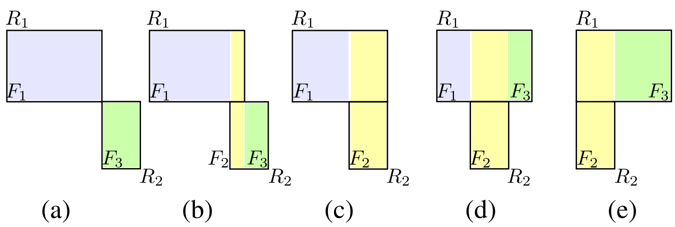


Figure 4: Description of rectangles F_1, F_2 and F_3 in the update procedure. When $R_1^R <_x R_2^R$ (a and b), $R_1 - F_2 = F_1$ and $R_2 - F_2 = F_3$, else $R_1 - F_2 = \{F_1, F_3\}$ (c, d and e). If $R_1^R =_x R_2^L$, $F_2 = \emptyset$, when $R_1^R =_x R_2^R$, $F_3 = \emptyset$ and finally $F_1 = \emptyset$ in the case $R_1^L =_x R_2^L$.

We show in the next section how we prune some nodes and arcs of the Reeb graph (and thus remove some k-arcs from the recoding of $\mathcal{E}$) so that the resulting irregular object is homotopic to a circle: this is indeed what we expect from a digital contour which is the boundary of a connected digital shape.

Guided by the Reeb graph, the computation of the polygonal representation of $\mathcal{E}$ is then performed by vectorizing independently each remaining k-arc.

3. Topological reconstruction of a noisy contour

We now propose to analyze noisy digital contour by using Kerautret and Lachaud's local noise detector [17]. This is a method for estimating locally if the digital contour is damaged, what is the amount of degradation and what is the finest resolution at which this part of the contour could be considered as noise-free. This part of the contour is then covered by a pixel whose size is the resolution determined by the above-mentionned noise detector. The higher the amount of noise is, the biggest the pixels are. In Figure 5-(b,g), we show an example of the output of this parameter-free algorithm applied to the two noisy digital objects depicted in Figure 5-(a,f).

As shown in Figure 5-(b,g), the resized pixels overlap and thus cannot be viewed as an irregular isothetic object (Definition 1). However each resized pixel contains a given number of pixels (at the initial resolution) so that the set of resized pixels covers a subset of the input image. This subset, which is an irregular isothetic object, is transformed into a new one, whose cells are of increasing left border (see Section 2.2). It is in turn filtered before the polygonal reconstruction.

The input digital contour is the interpixel contour of a 4-connected set of pixels. Since it is the boundary of a simply connected shape, it is expected to be homeomorphic to a circle. However, as in [26], the set of resized pixels and the resulting irregular object may not be homotopic to the input digital contour nor to a circle. It may contain either no hole or more than one hole. Thanks to the Reeb graph, which encodes the topology of the irregular object, we can decide whether we are in a general case (one hole) or not (none or more than one hole). If there is no hole, the filtering procedure is stopped and the set of resized pixels is computed again, but with a lower maximal resolution in the noise detector (parameter n in [17]). This filtering procedure is repeated until one hole is detected or until the resolution reaches the one of the input image. The latter case can happen only for one pixel-wide digital contours, which is not a reasonable input for a shape boundary and which can be independently processed.

In the general case, we choose not to process the k-arcs associated to the Reeb graph arcs that do not belong to the cycle. Thus the polygonal reconstruction is expected to be a simple closed polygon, for which exactly two edges are incident to each vertex. For instance, only reconstructing the k-arcs associated to the Reeb graph arcs of the (unique) cycle in Figure 5-(d) is a way of avoiding extra polygonal lines in the k-arcs pointed by arrows in Figure 5-(c).

The filtering procedure consists in two steps. First, we remove all degree-one nodes and their incident edges. This removes all sub-trees in the graph. Either the remaining subgraph is empty (no hole) or there is only one connected set of nodes whose degrees are each greater than two (at least one hole). (See the first part of Algorithm 1). If the procedure leads to a graph with no hole, then it means that the processed irregular object does not contain any hole. In this case, we loop back to the multi-scale noise detector and re-run it with a lower maximal resolution. This iterative process that progressively decreases the maximal resolution is illustrated in Figure 6.

Next, in order to get an irregular object that is homotopic to the initial digital contour, we remove internal connections, *i.e.* arcs whose terminating nodes have a degree strictly greater than two (see second row of Figure 5 and end of Algorithm 1). If the procedure leads to a graph with several connected components, then it means that the processed irregular object contains very thin parts. In this case, we loop back to the multi-scale noise detector and re-run it with a lower maximal resolution. This iterative process is illustrated in Figure 7.

The whole filtering process is illustrated in Figure 5-(h-i) and the proof of the correctness of Algorithm 1 is given in Appendix A. In the next section, we describe and compare several methods to vectorize the resulting irregular object, so as to get a simple closed polygon Figure 5-(j).

Algorithm 1: Filtering process.

input : $\mathcal{A}$, the sequence of n k-arcs recoding $\mathcal{E}$, ordered according to the left and top border of their first cell. $\mathcal{G}$, its associated Reeb graph.
output: $\mathcal{A}, \mathcal{G}$ are modified in order to obtain a single cycle in $\mathcal{G}$. The procedure returns true if $\mathcal{G}$ contains one and only one cycle, false otherwise
var : Q is a queue of nodes.

```
for each node (x) ∈ G do                              {Push nodes in queue}
    push (x) in Q ;
while Q is not empty do                 {Removing possible external nodes}
    (x) ← top of Q ;
    pop Q ;
    if Deg (x) = 1 then
        e ← (x)-(y): the arc in G connected to (x);
        if Deg (y) ≥ 2 then
            a ← the k-arc in A associated with e ;
            remove a from A ;
            remove (x) from G ;
            remove e from G ;
            push (y) in Q ;
if G has only one node then
    return false ;
for i from 0 to n − 1 do                   {Removing internal connections}
    e = (u)-(v): the arc in G associated to the k-arc a_i ∈ A ;
    if Deg (u) > 2 ∧ Deg (v) > 2 then
        remove a from A ;
        remove e from G ;
Make a breadth-first search in G ;
if the number of visited nodes is equal to the number of nodes in G then
    return true ;
else
    return false ;
```

4. Unsupervised polygonalization of noisy digital contours

Guided by the pruned Reeb graph, the computation of the polygonal representation of $\mathcal{E}$ is performed by reconstructing

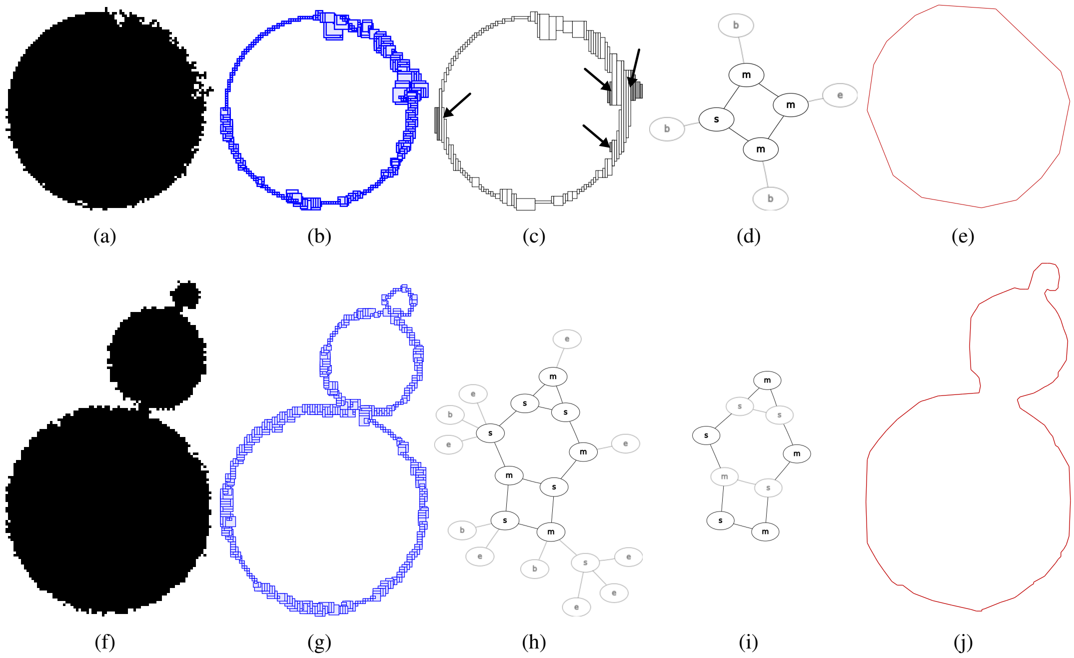


Figure 5: From a noisy contour (a), we build a set of resized pixels (b). Then, we filter the result of our vectorization algorithm by removing k-arcs that do not belong to the polygonal minimal contour (the ones pointed by arrows). To do so, we remove their associated edges in the Reeb graph (d), which lead to the desired polygonal contour (e). More complex topologies may also be considered, thanks to two more passes in our filtering procedure (h,i), which create a valid polygonal contour (j).

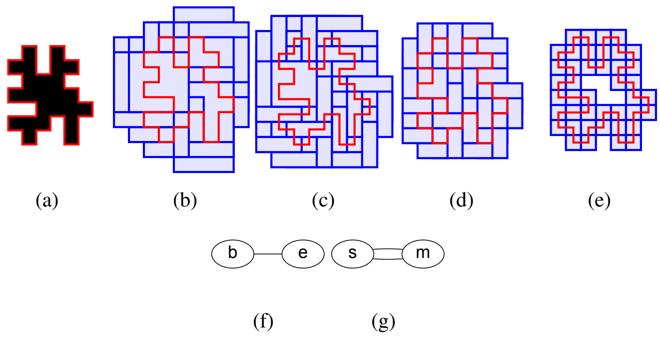


Figure 6: The multi-scale detector applied to the digital object (a) leads to an irregular object that does not contain any hole (b). Since the Reeb graph has only a single edge (f), our filtering procedure is able to detect this anomaly. We loop back with the detector, which is run with a lower maximal resolution (c). For instance, we have to loop again twice (d,e) to obtain a valid object with a (tiny) hole inside it (g).

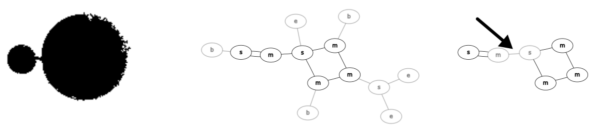


Figure 7: In this example, the Reeb graph is first pruned by removing external nodes (b). Then, the second phase removes the pointed edge in the graph, which leads to a disconnection (c). In this case, we loop back to the noise detector like in the case where no hole is detected (see Figure 6).

independently each remaining k-arc. In order to easily glue together each polygonal line into one global structure, each polygonal line is set to begin at the center of the first cell and to end at the center of the last cell of the vectorized k-arc. Between these two points, any polygonal line is valid. But among them, we are looking for the one that represents the most faithfully the k-arc (and the underlying unknown shape). It is reasonable to think that this polygonal line must belong to the set of polygonal lines that entirely lie within the k-arc. That is why most of the techniques we present below check if the computed polygonal line passes through the intersections between two successive k-adjacent cells. Due to the construction of the k-arcs (see Section 2.2), these intersections are vertical straight segments (possibly degenerated as a point) of increasing x-coordinate: they are called *input ranges*. Their extremity of greatest (resp. smallest) y-coordinate is called *upper* (resp. *lower*) *input point*.

In the subsections below, we recall the vectorization techniques of [34] and [37] before introducing a new method that minimizes the length of the resulting polygonal line.

4.1. Greedy decomposition into visibility cones

This method, introduced in [34] and inspired from [28], is driven by an iterative construction of a visibility cone (VC for visibility cone). For instance, in Figure 8-(a), a simple k-arc is decomposed into two visibility cones, which leads to a polygonal line composed of two segments.

The method can be roughly described as follows. We first initialize the cone apex with the center of the first cell and its base with the lower and upper points of the first input range. Then, the cone is updated for each new input range so that there is at least one ray coming from the cone apex and passing through all the input ranges. When a new input range cannot be visible from the cone apex, a new cone is set up, and its apex is added to the polygonal line. This point is the middle of the intersection between the bisector of the previous cone and the last scanned cell.

Even if this algorithm is linear-time, it is a greedy approach that could lead to some very short segments and acute angles (Fig. 8-(a)) This is why two other solutions have been proposed in [37]. We recall them in the next subsection.

4.2. Greedy decomposition into straight parts

The two methods we present here consist in decomposing a given k-arc A into *straight parts*, *i.e.* sets of k-adjacent cells that can cover a straight line (see for instance Fig. 8-(b)). In the general case, there are infinitely many straight lines that pass through a straight part and the union of all the transversal lines is called *preimage*. Note that all the transversal lines are considered here and not only those passing through a given point as in the visibility cone approach of the previous subsection. We use O'Rourke's algorithm [23] to incrementally decide whether A is straight or not and compute its preimage. Once a straight part has been detected, if the last cell of this part is not the last cell of A, we start the recognition of a new straight part and so on. The whole process is linear-time. In Figure 8, a simple k-arc is decomposed into two straight parts whose preimage is drawn in light blue.

A first and simple approach to transform this decomposition into a polygonal line is to link, for each straight part, the straight lines passing through the middle of the first and last input ranges. We call this method S2 (Simple and Straight). Even if the resulting polygonal line may be partly out of the k-arc (Figure 8-(b)), this is an interesting way of decomposing a k-arc because the resulting polygonal line contains few segments and preserves the straight parts.

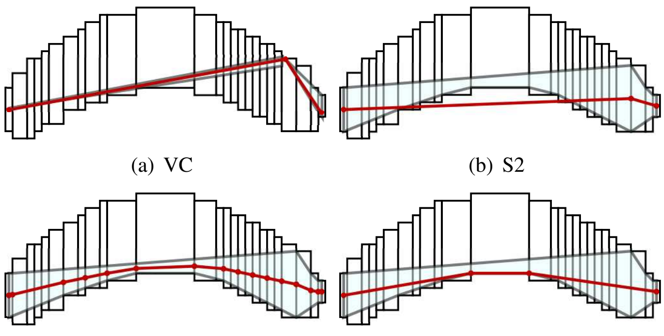
However, in order to get a polygonal line that entirely lies within the k-arc, we propose another solution that takes into account the shape of the preimage of each straight part (C2, meaning Complex and Curved).

The preimage is implicitly described by some consecutive vertices of the lower (resp. upper) part of the convex hull of the upper (resp. lower) input points. The idea is to incrementally compute the polygonal line that is lying in the middle of the preimage. More precisely, for each input range whose upper or lower input point belongs to the preimage, a new vertex is set to the middle of the intersection between the preimage and a vertical line passing through the input points (see [37] for more details).

This method leads to smooth polygonal lines (small angle variations between two consecutive edges) that are well centered within the k-arc (Figure 8-(c)). However they usually have a lot of segments, and their geometry does not necessarily reflect the local convexity or concavity of the underlying shape (see for instance the last part of the polygonal line depicted in Figure 8-(c)).

4.3. Minimal length polygonal line

In this subsection, we propose to compute, among the set of polygonal lines that entirely lie within a given k-arc A, the one of minimal length that joins the centers of the first and last cells of A. This polygonal line, which always exists and which is unique (when there are no collinear vertices), has been introduced in [21, 30] as the *minimal length polygon* (MLP). Its n-dimensional version is known as *relative convex hull* [31]. In the case of input ranges of increasing x-coordinate, the MLP is nothing else than a sequence of upper or lower parts of convex hulls. Since the input points are sorted according to the x-coordinate, their computation can be incremental and linear-time due to the simple Andrew's monotone chain algorithm [2]. Figure 8-(d) illustrates the MLP reconstruction of a k-arc.



(a) VC (b) S2

(c) C2 (d) MLP

Figure 8: The four versions of polygonalization on a single k-arc. Output of the VC (visibility cone) method in (a). Output of the S2 (simple and straight) method in (b), C2 (complex and convex) method in (c), both based on the preimage of each straight part. MLP (minimum length polygon) in (d).

The method uses a visibility cone whose apex is always a MLP vertex. We first initialize the cone apex with the center of the first cell and its base with the lower and upper points of the first input range. Then, the lower (resp. upper) part of the convex hull of the upper (resp. lower) input points are incrementally computed and the cone is updated while there is at least one ray coming from the cone apex and separating the two convex hulls. When a new input range is located strictly above (resp. below) the visibility cone, the apex of a new cone is set to the vertex of maximal x-coordinate of the lower (resp. upper) convex hull that is visible from the upper (resp. lower) point of the new input range. All the vertices of the lower (resp. upper) convex hull scanned during this update process are stored in the MLP vertices list.

The resulting polygonal line reflects well the local convexity or concavity of the underlying shape. It not only minimizes its length but also the number of its inflection points, hence it is rather smooth.

4.4. Comparison and discussion

We show in Figure 9 an example of the previous vectorization techniques on two irregular objects: one is the result of a quadtree decomposition, the other one uses the multi-scale noise detection.

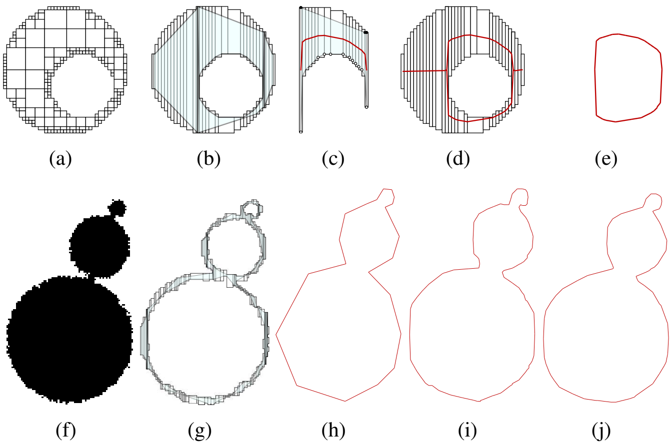


Figure 9: Illustration of our contribution on an object digitized with a quadtree (a). (b) is the complete preimage computed on each k-arc encoding the object. One could note that the k-arc at the bottom is decomposed into two straight k-arcs. In (c), we present the reconstruction of a single k-arc, and the associated preimage and upper/lower convex hull points. We also depict the complete polygonal reconstruction of the object, constructed inside the preimage (d), and the final contour obtained with our filtering procedure explained in the previous section. We also show the computation of the complete preimage (g) for the noisy contour (f), and the final reconstructions S2 (h), C2 (i), MLP (j).

All four presented methods have a linear-time complexity. However they yield different polygonal reconstructions. The differences are summed up in Table 1. Unlike the other methods, S2 does not lead to a polygonal line that stays within the k-arc, but it respects the straight parts. MLP and C2 lead to smoother polygonal lines than VC (MLP leads to the smoothest polygonal line since it minimizes its length). The polygonal line computed from C2 is the most centered within the k-arc, but the MLP is unique and respects the convex and concave parts. Note that none of the methods minimizes the number of segments, even if the S2 method usually yields smaller polygonal line (see next section).

Criteria	VC	S2	C2	MLP
Is unique	no	no	no	yes
Stays inside the k-arc	yes	no	yes	yes
Is centered within the k-arc	no	no	yes	no
Respects the straight parts	no	yes	no	no
Respects the convex & concave parts	no	no	no	yes
Minimizes the length/angle variation	no	no	no	yes
Minimizes the number of segments	no	no	no	no

Table 1: Theoretical comparison of the four proposed polygonalization methods.

5. Experimental results

5.1. Comparative study

To experiment the quality of the proposed algorithms, we first consider a polygonal shape that was perturbed by a Gaussian noise, with different standard deviations ($\sigma_0 = 0$, $\sigma_1 = 75$, $\sigma_2 = 125$, $\sigma_3 = 175$). These images were generated with two different grid sizes $h = 1$ and 0.5 (Figure 10 (a,g)). The

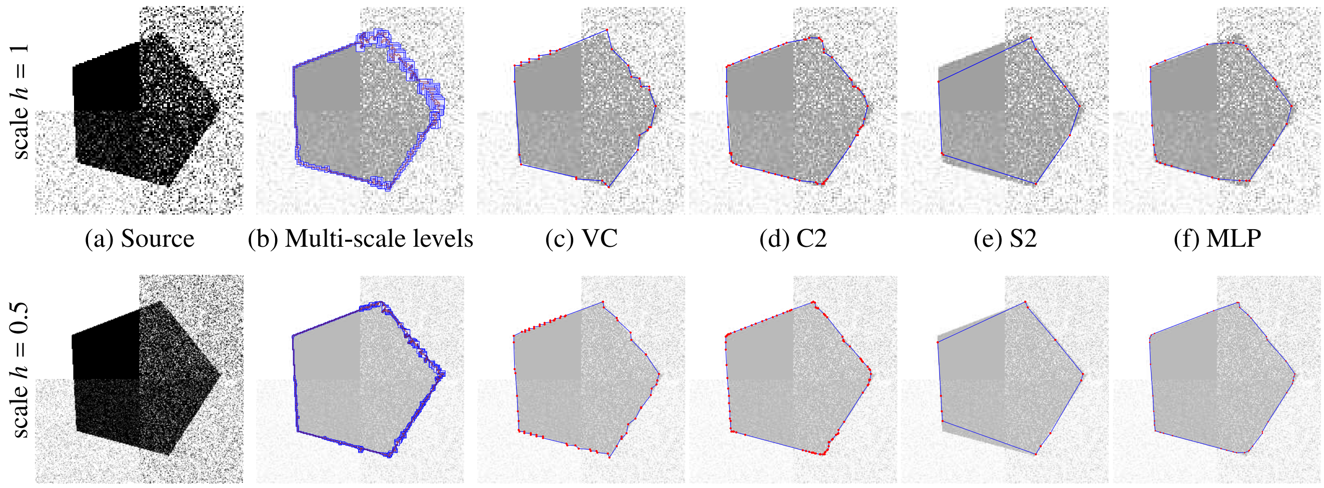
		VC	C2	S2	MLP
$h = 1$	n	33	60	**9**	31
	E_d	**0.80**	0.83	2.73	1.25
	θ^2_{err}	0.12	0.06	0.07	**0.05**
$h = 0.5$	n	69	91	**12**	41
	E_d	**0.74**	0.65	2.99	0.78
	θ^2_{err}	0.12	0.04	0.05	**0.02**

Table 2: Error measures from contour reconstructions of Figure 10. The mean minimal euclidean distance (E_d) and error on tangent orientations (θ^2_{err}) were computed for each algorithms version on different scales h.

resized pixels (illustrated on images of Figure 10 (b,h)) were obtained from the digital contours extracted by using a simple threshold (set to 128) (images (b,h)) and boundary tracking algorithm. In order to measure the resulting quality of the four reconstructions illustrated on images (c-f) and (i-l) we applied various measures given on Table 2. These measures are the total number of points (n), the mean minimal euclidean distance (E_d) between the source contour points P_i to the resulting polygon, and the error on tangent orientations (θ^2_{err}). The measure E_d was obtained after associating each contour points P_i of the initial shape (non noisy) to the nearest consecutive vertex pair V_k, V_{k+1}. These associations were also used to determine the tangent error θ^2_{err} where θ_{err} is the angle between the tangent vector defined from V_k, V_{k+1} and the tangent provided by the $\lambda - MST$ estimator [18] applied on the source (undamaged) discrete contour.

The experiments confirm the awaited improvements provided by the Algorithm C2 in comparison with the use of the algorithm based on visibility cone [34] (denoted as Alg-VC). It is visible especially for the tangent error measure θ^2_{err} but also for the distance error E_d. The second variant Algorithm S2 produces a more compact representation while preserving a moderate tangent error θ^2_{err}. However this last algorithm is less convenient on the point of view of the E_d error. On the point of view of the tangent error measure θ^2_{err}, the algorithm MLP appears to give the best results on each image size.

Finally, we compare our methods with algorithms developed by Nguyen and Rennesson [22] which are based on a global optimization scheme in association with the Marji's criteria (MC) or another one proposed by the authors (NC). In Figure 11, we present the polygonal contour obtained from our methods, and from the NC and MC algorithms which are both parameter free approaches. For each experiment, we measure the Hausdorff error (δ_H) and the previously described errors (see Table 3). The comparisons show that the proposed approaches are less compact than both the NC or MC but provide better precision for the δ_H and E_d errors. On the point of view of the tangent orientation error θ^2_{err} our approaches with C2 or S2 are comparable with the one of the NC algorithm, while MLP outperforms all. Other complementary comparisons were also performed with two recent parametric methods. The first one is the polygonal reconstruction from the visual curvature [19] which uses a parameter associated to the scale of the contour analysis. The second one exploits another way to take into accounts the noise



(a) Source (b) Multi-scale levels (c) VC (d) C2 (e) S2 (f) MLP

(g) Source (h) Multi-scale levels (i) VC (j) C2 (k) S2 (l) MLP

Figure 10: Illustration of the reconstruction algorithms applied on different image scale. The images (b,h) show the multi-scale levels obtained from the source contours (a,g). The reconstructed polygons associated to Alg-VC (that uses previous work), C2, S2, MLP are given respectively on (c-f) and (i-l). Geometric measures are give on Table 2.

	VC	C2	S2	MLP	Ngu09 [22]	Marji [22]
n	211	457	100	212	**52**	24
δ_H	**6**	6.07	8.92	6.34	10.81	10.63
E_d	0.757	**0.713**	1.236	0.842	1.221	2.878
θ^2_{err}	0.130	0.076	0.071	**0.040**	0.062	0.131

	Siv11 [29]		Liu08 [19]	
	$d = 3$	$d = 5$	$s = 0.01$	$s = 0.03$
n	157	85	176	75
δ_H	9.98	8.544	11.401	11.401
E_d	1.068	1.808	0.859	1.917
θ^2_{err}	0.103	0.104	0.128	0.0619

Table 3: Geometric measures of the reconstructed shapes of Figure 11. The different proposed algorithms (four first columns of first tabular) can be compared with other parameter free approaches [22] (two last columns of first tabular). The second tabular gives measures obtained from recent parametric approaches for comparisons [29, 19].

by using the the Fréchet distance defined between the initial discrete contour and the resulting polygon [29]. We apply the reconstructions with several parameter settings illustrated on Figure 11 (i-l). The parameters were first manually tuned to favour the closeness to initial data with some noisy areas on the last quadrant ($d = 3$ $s = 0.01$) and the second one gives the priority to the noise removal ($d = 5$, $s = 0.03$). The measure of Table 2 confirms the performance of the proposed methods since the MLP based algorithm outperforms all the geometric measures for all set of parameters.

5.2. Complex image analysis

The Algorithm C2 was also experimented on real images of characters, acquired from a photographed document. A given threshold was used to extract the digital contours on which the resized pixels were computed (as illustrated on the second row Figure 12). We thus show that our vectorization algorithm could be applied in document analysis systems.

Our algorithms may also be used in the polygonal modeling of region of interest in many image analysis applications. Here, we depict the extraction of a part of an heart in a MRI (Magnetic Resonance Imaging) in Figure 13. Despite of the presence of noise in the image, we are able to propose a clean reconstruction of the selected region.

We also present a last application of our work in a project of leaf recognition for smartphones.[1] In this context, leaves may be detected in very complex environments by computing a distance map with Gaussian mixture models [4, 5]. Thanks to this map, we are able to compute a polygonal model of the leaf, even if the background color model is very close to the one of the treated object (see Figure 14).

5.3. Adaptive polygonalization by combined curved/flat reconstructions

Here, we propose to combine the two versions S2 and C2 we have introduced before in order to adaptively reconstruct noisy shapes. The meaningful scale detection we use [17] is able to distinguish curved and flat parts of the input noisy contour. In Figure 15-(a,g), we show extracted resized pixels of a digital contour. In our system, for each k-arc, we count the number of flat and curved included inside it, respectively n_f and n_c. We then determine that this k-arc is curved if we have:

$$\frac{n_c}{n_c + n_f} \geq \eta, \qquad (1)$$

where $\eta \in [0, 1]$ is a given threshold. In this case, we apply the C2 version, and S2 one otherwise. We give in Figure 15 some examples of reconstructions with various values for η, for two images.

[1] `http://liris.cnrs.fr/reves/content/en/index.php`

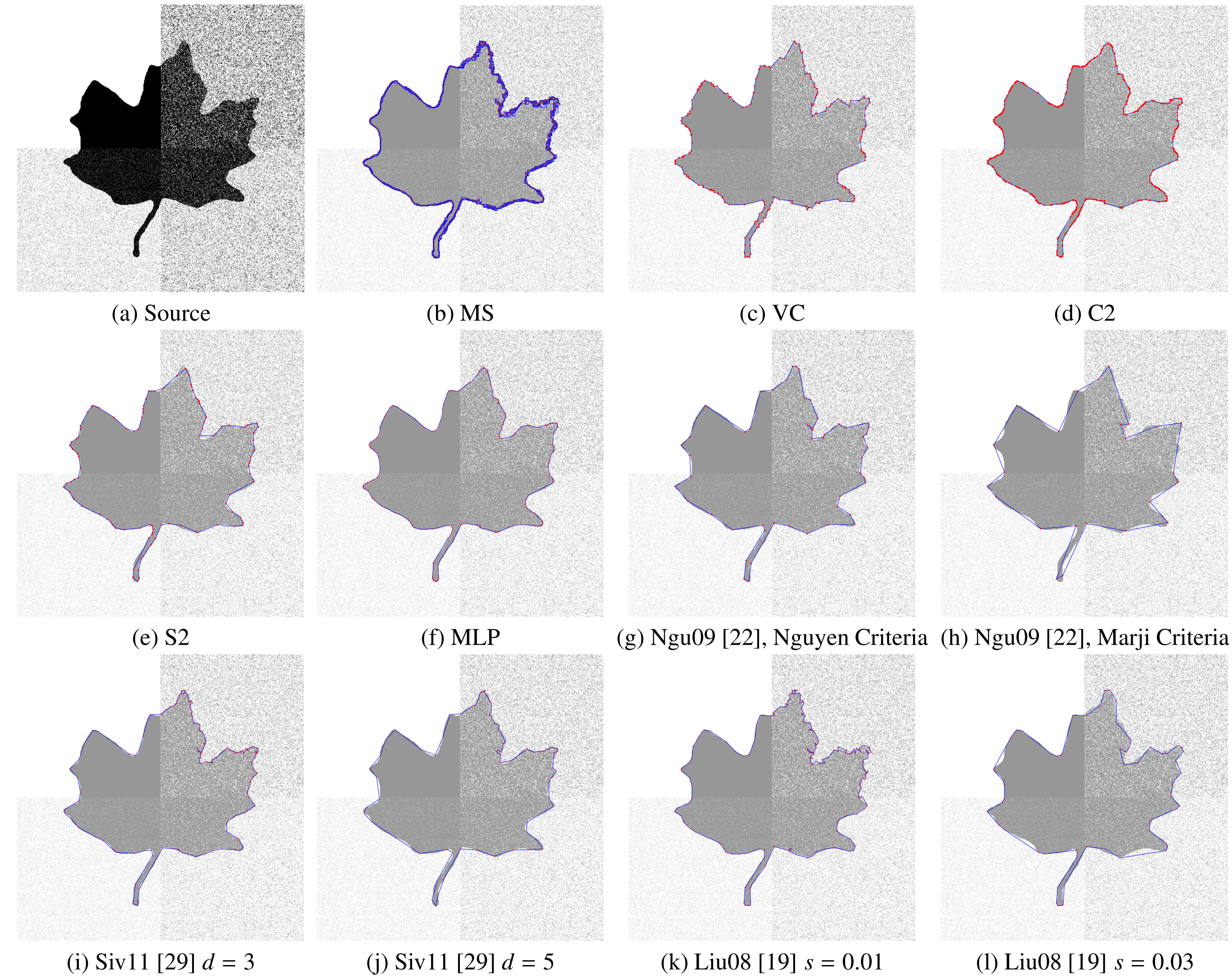


Figure 11: Comparisons of the proposed approaches (b-f) with others recent parameter free approaches [22] (g,h) and with parametric approaches (j-l) [29, 19]. Detailed comparisons on geometric measures are given on Table 3.

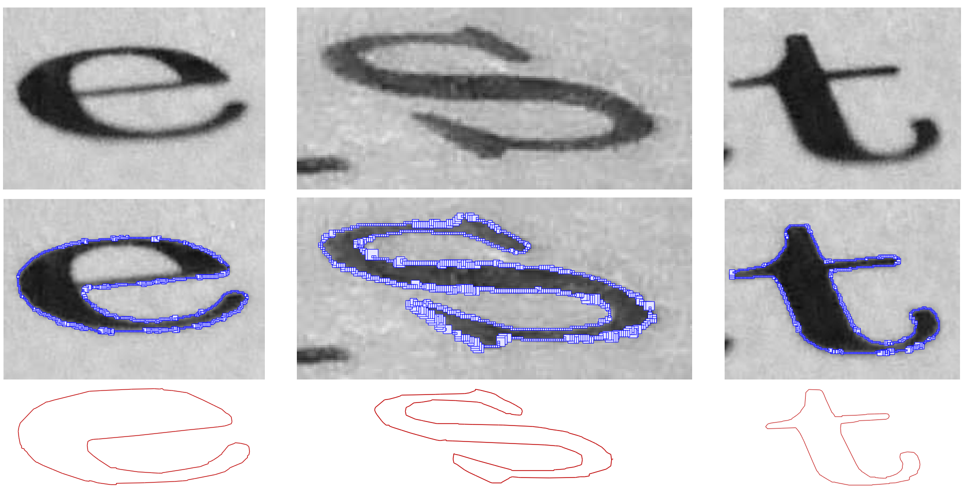


Figure 12: The meaningful boxes extracted from scanned characters (center), and the final reconstruction we propose with C2 (bottom).

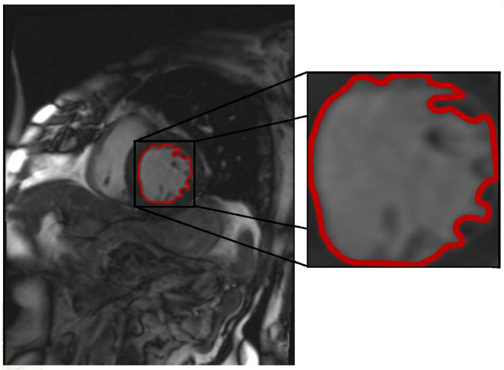


Figure 13: Extraction of a region of interest in MRI of heart with C2.

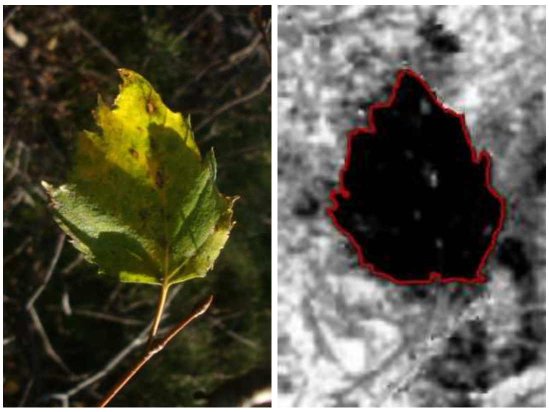


Figure 14: Extraction of the aspen leaf from the background and construction of a precise polygonal model of it.

6. Conclusion and Future Works

In this paper, we address the problem of vectorization of noisy digital contours. We transform the resized pixels obtained by Kerautret and Lachaud's algorithm [17] into an irregular isothetic object recoded in a set of k-arcs whose topology is stored into a Reeb graph. We first show how to use the Reeb graph in order to prune the set of k-arcs so that it is homotopic to the initial digital contour. Then we review different geometrical algorithms (VC, S2, C2), and propose a new one (MLP), in order to build a polygonal representation of each k-arc. The resulting polygonal structure is obtained by gluing together the independent polygonal lines. The whole polygonalization process takes a linear-time in the number of cells. We have shown in the experiments that our proposals are very efficient w.r.t. to several other techniques of the literature. We have also presented applications in image analysis that reveal the interest of our system, and an original way to combine two complementary methods of polygonalization (S2 and C2).

We plan to work on noisy 3-D surfaces, and develop a complete framework in a similar way as the one presented in this article. We thus have to adapt the noise detector in order to compute a multi-scale representation of the input object. Then, we would like to compute the Reeb graph, and use this topological tool to guide an original polyhedrization algorithm that process overlapping irregular 3-D cells.

Appendix A. Proof of the correctness of the Reeb graph filtering procedure

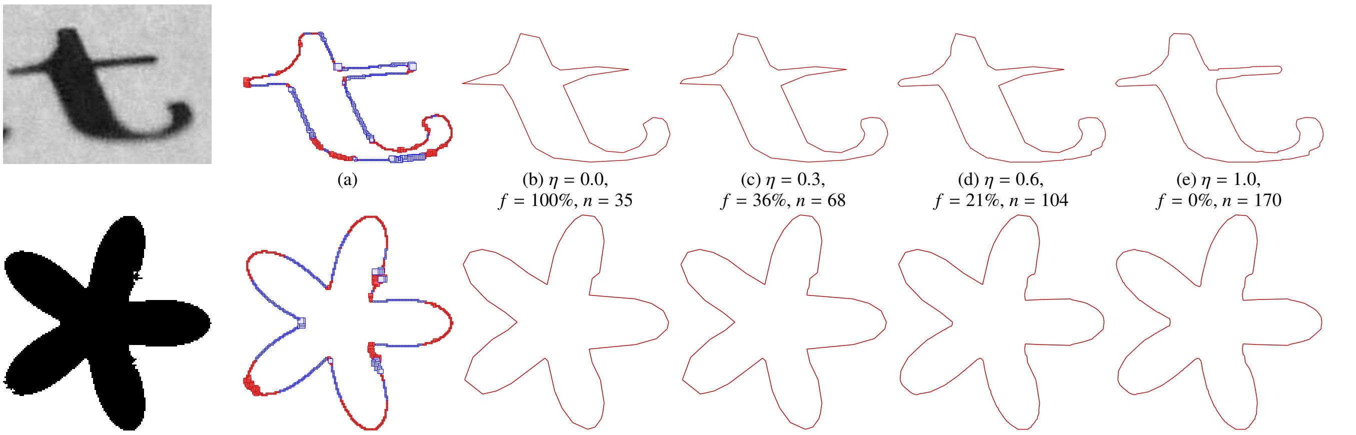
Lemma 1 (Validity of Algorithm 1)**.** *Algorithm 1 returns true if the filtering process yields a subgraph that contains one and only one cycle, but false otherwise.*

Proof. Algorithm 1 consists in two steps. The first one iteratively removes nodes of degree one and their unique indicent arcs (i). The second one iteratively removes arcs incident to two nodes of degree strictly greater than two (ii). Let us see what is the impact of these two steps on the graph structure.

(i) A the end of the first step, since the input graph is connected, only two cases may occur: either there is only one node (of degree zero), or there is a connected set of nodes (of degree greater than or equal to two). The first case occurs only if the input graph is a tree (a connected graph without any cycle). This can be shown by structural induction. The base case is any tree of only one node. Then, connecting with a new arc, a new node to any node of a tree yields a tree bigger of one node and one arc, because no cycle has been created. Due to the previous result, it is clear by contradiction that the second and last case occurs only if the input graph has one cycle or more.

In the first case the algorithm stops and returns false, otherwise it performs the second step in order to keep only one cycle.

(ii) If the resulting graph is not connected after the second step, the algorithm retuns false. Otherwise, we prove below that it returns true because it contains one and only one cycle.



(a) (b) $\eta = 0.0$, $f = 100\%$, $n = 35$ (c) $\eta = 0.3$, $f = 36\%$, $n = 68$ (d) $\eta = 0.6$, $f = 21\%$, $n = 104$ (e) $\eta = 1.0$, $f = 0\%$, $n = 170$

(f) (g) $\eta = 0.0$, $f = 100\%$, $n = 53$ (h) $\eta = 0.1$, $f = 70\%$, $n = 55$ (i) $\eta = 0.6$, $f = 25\%$, $n = 173$ (j) $\eta = 1.0$, $f = 0\%$, $n = 257$

Figure 15: From the curved/flat feature extracted (red:curved, blue:flat) with the multi-scale detector (a,g), we propose an adaptive reconstruction with several values for η. For each case, we also give the final percentage of flat k-arcs f, and the number of points in the reconstruction n.

Due to the construction of the Reeb graph according to the order $\preceq^L$, after the removal of all degree one nodes, there is at least one minimal node $s^\star$ and one maximal node $m^\star$ in the resulting graph. In the initial Reeb graph, there is a tree rooted at $s^\star$ (resp. $m^\star$), which contains all the nodes smaller (resp. greater) than $s^\star$ (resp. $m^\star$), and which is removed during the first step. Otherwise $s^\star$ (resp. $m^\star$) is not the minimal (resp. maximal) node of the resulting graph, which raises a contradiction. This means that $s^\star$ and $m^\star$ are both of degree two after the first step.

As a consequence, at the end of the second step, the set of connected nodes contains at least two nodes of degree two ($s^\star$ and $m^\star$), but no node of degree strictly greater than two (removed). Since there is no node of degree strictly less than two (due to the first step), there is extactly one cycle, which concludes the proof. □

References

[1] Anagnostopoulos, C.-N.E., Anagnostopoulos, I.E., Psorulas, I.D., Loumos, V. and Kayafas, E.: License plate recognition from still images and video sequences: a survey. *IEEE Trans. on ITS*, **9**(3):377–391, 2008.

[2] Andrew, A.M. Another efficient algorithm for convex hulls in two dimensions. *Information Processing Letters*, **9**(5):216–219, 1979.

[3] Canny, J.: A computational approach to edge detection. *IEEE Trans. on PAMI*, **8**(6):679–698, 1986.

[4] Cerutti, G., Tougne, L., Vacavant, A. and Coquin, D.: A parametric active polygon for leaf segmentation and shape estimation. In *Proc. of ISVC*, Springer LNCS 6938, pp. 202-213, 2011.

[5] Cerutti, G., Tougne, L., Mille, J., Vacavant, A. and Coquin, D.: Guiding Active Contours for Tree Leaf Segmentation and Identification. In Proc. of CLEF, 2011.

[6] Coeurjolly, D. and Tougne, L.: Digital straight line recognition on heterogeneous grids. In *Proc. of SPIE Vision Geometry XII*, volume 5300 pp. 108–116, 2004.

[7] Coeurjolly, D. and Zerarga, L.: Supercover model, digital straight line recognition and curve reconstruction on the irregular isothetic grids. *Comp.& Graphics*, **30**(1):46–53, 2006.

[8] Cordella, L.P. and Vento, M.: Symbol recognition in documents: a collection of techniques? *Int. Journal on Document Analysis and Recognition*, **3**(2):73–88, 2000.

[9] Daniels, K.M., Milenkovic, V.J., and Roth, D.: Finding the Largest Rectangle in Several Classes of Polygons. Technical Report TR-22-95, *Center for Research in Computing Technology, Harvard University*, 1995.

[10] Davis, L. S.: Edge detection techniques. *Computer Graphics & Image Processing*, **4**:248–270, 1995.

[11] Debled-Rennesson I., Feschet F. and Rouyer-Degli J.: Optimal blurred segments decomposition of noisy shapes in linear time. *Comp.& Graphics*, **30**:30–36, 2006.

[12] Faure, A. and Feschet, F.: Linear decomposition of planar shapes. In *Proc. of IEEE ICPR* pp. 1096–1099, 2010.

[13] Graham, R.L. An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, **1**:132–133, 1972.

[14] Hoang, T. V.; Barney Smith, E. H. and Tabbone, S.: Edge noise removal in bilevel graphical document images using sparse representation in *IEEE International Conference on Image Processing - ICIP'2011*.

[15] Hilaire, X. and Tombre, K.: Robust and accurate vectorization of line drawings. *IEEE Trans. on PAMI*, **8**(4):890–904, 2005.

[16] Keil, J.M.: Polygon Decomposition. In *Handbook of Computational Geometry*, chapter 11, Elsevier Science, pp. 491–518, 2000.

[17] Kerautret, B., Lachaud, J.O.: Multi-scale analysis of discrete contours for unsupervised noise detection. In *Proc. of IWCIA* , Springer LNCS 5852, pp. 187–200, 2009.

[18] Lachaud, J.O., Vialard, A., and de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours, *IVC*, **25**(10):1572–1587, 2007.

[19] Liu, H., Latecki, L. J. and Liu W.: A Unified Curvature Definition for Regular, Polygonal, and Digital Planar Curves. *International Journal of Computer Vision*, **80**:104–124, 2008.

[20] Melkman, A.A.: On-line construction of the convex hull of a simple polyline. *Information Processing Letters*, **25**(1):11–12, 1987.

[21] Montanari, U.: A note on minimal length polygonal approximation to a digitized contour. *Communications of the ACM*, **13**(1):41–47, 1970.

[22] Nguyen, T.P. and Debled-Rennesson, I.: Parameter-free method for polygonal representation of the noisy curves. In *Proc. of IWCIA*, RPS, 2009.

[23] O'Rourke, J.: An on-line algorithm for fitting straight lines between data ranges. *Communications of the ACM*, **24**(9):574–578, 1981.

[24] O'Rourke, J. and Tewari, G.: The Structure of Optimal Partitions of Orthogonal Polygons into Fat Rectangles. *Computational Geometry: Theory and Applications*, **28**(1):49–71, 2004.

[25] Reeb, G.: Sur les points singuliers d'une forme de pfaff complètement intégrable ou d'une fonction numérique. *Comptes Rendus de L'Académie ses Sciences*, Paris 222, pp. 847–849, 1946.

[26] Rodriguez, M., Largeteau-Skapin, G. and Andres, E. Adaptive pixel resizing for multiscale recognition and reconstruction. In *Proc. of IWCIA*, Springer LNCS 5852, pp. 252–265, 2009.

[27] Rodriguez, M., Largeteau-Skapin, G. and Andres, E. Adaptive pixel size

reconstruction with topological control. In *Proc. of IWCIA*, RPS Progress in Combinatorial Image Analysis, 2009.

[28] Sivignon, I., Breton, R., Dupont, F. and Andres, E.: Discrete analytical curve reconstruction without patches. *IVC*, **23**(2):191–202, 2005.

[29] Sivignon, I.: A near-linear time guaranteed algorithm for digital curve simplification under the Fréchet distance. In *Proc. DGCI*, Springer LNCS 6607, pp. 333–345, 2011.

[30] Sklansky, J., Chazin, R.L., Hansen, B.J.: Minimum perimeter polygons of digitized silhouettes. *IEEE Transactions on Computers*, **21**(3):260–268, 1972.

[31] Sloboda, F., Zatko, B.: On approximation of jordan surfaces in 3D. In: *Bertrand, G., Imiya, A., Klette, R. (eds.) Digital and Image Geometry*. Springer LNCS 2243, pages 365–386. Springer, 2002.

[32] Tombre, K.: Analysis of engineering drawings: state of the art and challenges. In *Graphics Recognition Algorithms and Systems*, Springer LNCS 1389, pp. 257–264, 1998.

[33] Tombre, K. and Tabbone, S.A.: Vectorization in graphics recognition: to thin or not to thin. In *Proc. of IEEE ICPR*, pp. 91–96, 2000.

[34] Vacavant, A., Coeurjolly, D. and Tougne, L.: Topological and geometrical reconstruction of complex objects on irregular isothetic grids. In *Proc. of Int. Conf. of DGCI*, Springer LNCS 4245, pp. 470–481, 2006.

[35] Vacavant, A., Coeurjolly, D. and Tougne, L.: A framework for dynamic implicit curve approximation by an irregular discrete approach. *Graphical Models*, **71**(3):113–124, 2009.

[36] Vacavant, A.: Fast distance transformation on two-dimensional irregular grids. *Pattern Recognition*, **43**(10):3348–3358, 2010.

[37] Vacavant, A., Roussillon, T. and Kerautret, B.: Unsupervised polygonal reconstruction of noisy contours by a discrete irregular approach. In *Proc. of IWCIA*, Springer LNCS 6636, pp. 389–409, 2011.

[38] Thome, N., Vacavant, A., Robinault, L. and Miguet, S.: A cognitive and video-based approach for multinational license plate recognition. *Machine Vision and Applications*, **22**(2):389–407, 2011.

[39] Wenyin, L. and Dori, D.: A survey of non-thinning based vectorization methods. In *Proc. of Joint IAPR Workshops SSPR and SPR)*, Springer LNCS 1451, pages 230–241, 1998.

[40] Wenyin, L. and Dori, D.: From raster to vectors: extracting visual information from line drawings. *Pattern Analysis and Application*, **2**(1):10–21, 1999.