Unsupervised Polygonal Reconstruction of Noisy Contours by a Discrete Irregular Approach

Antoine Vacavant[1,⋆], Tristan Roussillon[2,3] and Bertrand Kerautret[4,5]

[1] Clermont Université, Université d'Auvergne, ISIT, F-63000, France
antoine.vacavant@iut.u-clermont1.fr
[2] Université de Lyon, CNRS
[3] Université Lyon 2, LIRIS, UMR5205, F-69676, France
tristan.roussillon@liris.cnrs.fr
[4] LORIA, UMR 7503 CNRS, Université de Nancy, France
[5] LAMA, UMR 5127 CNRS, Université de Savoie, F-73376, France
kerautre@loria.fr

Abstract. In this paper, we present an original algorithm to build a polygonal reconstruction of noisy digital contours. For this purpose, we first improve an algorithm devoted to the vectorization of discrete irregular isothetic objects. Afterwards we propose to use it to define a reconstruction process of noisy digital contours. More precisely, we use a local noise detector, introduced by Kerautret and Lachaud in IWCIA 2009, that builds a multi-scale representation of the digital contour, which is composed of pixels of various size depending of the local amount of noise. Finally, we compare our approach with previous works, by considering the Hausdorff distance and the error on tangent orientations of the computed line segments to the original perfect contour. Thanks to both synthetic and real noisy objects, we show that our approach has interesting performance, and could be applied in document analysis systems.

1 Introduction

The representation of graphical objects (such as symbols, line drawings, characters, *etc.*) with line segments is an important task for various document and image analysis applications. This vectorization stage has been widely studied since the 90's, and many algorithms have been designed [4, 6, 18]. Discrete or digital contours are natural outputs of image segmentation algorithms or digitization processes (*e.g.* document scanning). In most cases, digital contours are not perfect digitizations of ideal shapes but present noise and irregularities. In this case, classical approaches of contour detection generally need a parameter, and the output has to be filtered and post-processed (see Fig. 1 for an example with the Canny edge detector).

⋆ This work has been supported by the French National Agency for Research with the reference ANR-10-CORD-005 (REVES project).

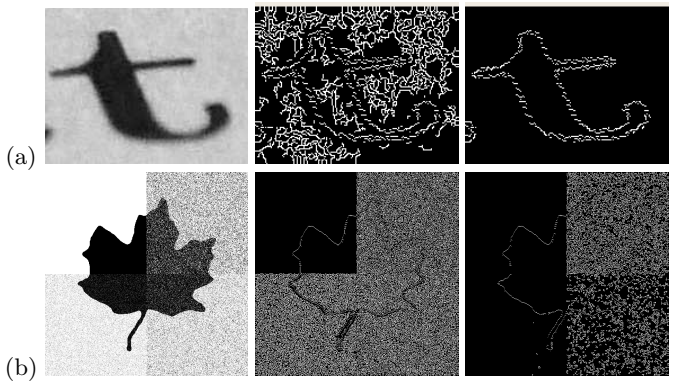


Fig. 1. The Canny edge detector applied on two images with two sets of paremeters. For image (a), even if we could obtain an interesting result, a post-process is necessary to filter the output of the detector in order to compute a linear contour. A very noisy image (b) cannot be efficiently handled by this detector, even with various parameters

Lately, two different approaches have been proposed in the digital geometry community. (i) The noisy digital contour (or a thick digital curve around it) is partitioned into thick (or blurred) segments [13, 5]. This last approach requires a global thickness parameter and thus cannot handle noises that are not uniform. (ii) To cope with this problem an adaptive pixel resizing method has been proposed in [12]. The idea is interesting but its implementation (as described in [12]) has several drawbacks. Firstly, the resizing function (which is not explicitly given) is based on the length of the symmetric tangents computed on the digital contour at the initial scale. Second, the resized pixels overlap so that the polygonalization is performed by a complex generalized preimage algorithm. Finally, the set of resized pixels may not be homotopic to the input digital contour and the topological control process proposed by the authors requires a skeleton computation.

In this paper, we propose a novel approach to compute a polygonal reconstruction from a noisy digital contour. For this purpose, we compute a set of resized pixels from a noisy digital contour thanks to a local noise detector [7]. The idea is to locally look at the length of the maximal digital straight segments lying on the input digital contour at decreasing resolutions. The resolution beyond which we observe that the evolution of the length of the maximal segments is similar to the theoretical behavior for digitizations of smooth shapes does not contain any noise. The pixels at this resolution corresponds to bigger pixels at the initial resolution so that the higher the amount of noise is, the biggest the pixels are.

This set of resized pixels is transformed into an irregular isothetic object composed of rectangular cells and whose topology is stored into a Reeb graph [15]. For the reconstruction, each arc is vectorized into a polygonal line. The polygonal

lines are then linked together so that the resulting polygonal structure reflects the topology of the irregular isothetic object.

In [15], arcs are vectorized following a visibility cone approach. Even if this reconstruction method has a linear-time complexity, it is a greedy approach that may induce an increasing error that leads to some very short segments and very acute turn angles. In this paper, we segment each arc into straight parts in linear-time using O'Rourke's algorithm [10], which is much simpler than the generalized preimage approach of [12]. We then propose two different reconstructions: (i) the first one takes into account the shape of the preimage of each straight part so that the resulting polygon lies within the irregular isothetic object, (ii) the second one is a simpler method, more convenient for objects that contain straight parts, but the resulting polygon may not completely lie within the irregular isothetic object.

In the next section, we recall definitions about irregular isothetic objects, and the previous work of [15]. We then describe our polygonalization methods and use it in order to reconstruct a noisy digital contour. Finally, we present several experiments and comparisons.

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 definition [3, 17].

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 non overlapping rectangular cells whose edges are parallel to the X and Y axes. 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 $(l_R^x, l_R^y) \in {\mathbb{R}_+^*}^2$.*

We say that two cells R_1 and R_2 are *ve*-adjacent if they share either a vertex or an edge, and *e*-adjacent if they share an edge. In a more general way, we say that R_1 and R_2 are k-adjacent, and k may be interpreted as e or ve in the following definitions. A set of cells $\mathcal{E}$ is a $k-$arc iff for each element of $\mathcal{E} = \{R_i\}_{1 \leq i \leq n}$, R_i has exactly two $k-$adjacent cells, except R_1 and R_n. A set of cells $\mathcal{E}$ is a $k-$object iff for each couple of cells $(R_1, R_2) \in \mathcal{E} \times \mathcal{E}$, there exists a $k-$arc between R_1 and R_2 in $\mathcal{E}$ (Fig. 2, left)).

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 $x_R - (l_R^x/2)$ and for sake of clarity we write it $R^L = x_R - (l_R^x/2)$.

Definition 2 (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 $\preceq^L$, based on the cells borders:*

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

This order relation is of great importance either for the Reeb graph computation or for the segmentation of each k-arc into straight parts using O'Rourke algorithm, which requires that the input ranges have increasing x-coordinate.

2.2 Previous Algorithm for Irregular Object Vectorization

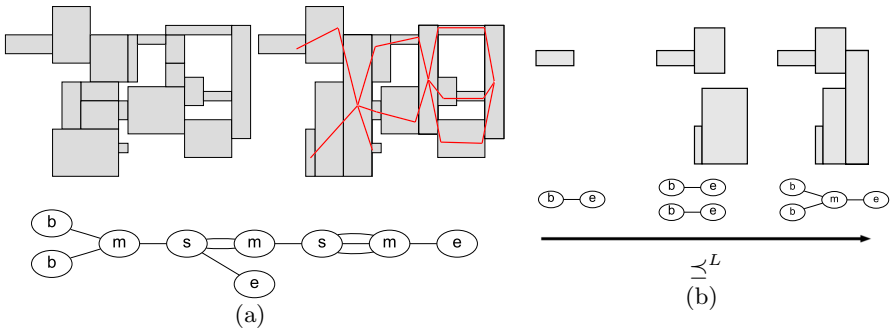


Fig. 2. (a) An example of an irregular object $\mathcal{E}$ *(left)*, the final recoded structure with arcs, the obtained polygonalization *(right)* and the Reeb graph associated to the order defined on $\mathcal{E}$ *(bottom)* [16]. In (b), we show the recognized $k-$arcs and the associated Reeb graph for some iterations of this algorithm, in respect to the $\preceq^L$ order.

The work of Vacavant *et al.* [15] aimed to develop an algorithm that vectorizes a k-object with line segments. This method is divided into two main steps.

Representation of the Topology of an Irregular Object. The Reeb graph [11] of the input irregular k-object $\mathcal{E}$ is a way of representing the topology of $\mathcal{E}$. The k-object $\mathcal{E}$ is scanned from left to right according to the order induced by $\preceq^L$, given in Definition 2 (see Fig. 2 for an example).

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 edge 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 edges as there are parts. Conversely, if two connected parts merge, we link the corresponding edges to a node (m for merge) (see Fig. 2).

Moreover, the initial set of cells is recoded into a new one (without changing the shape of the object however) so that each edge of the Reeb graph corresponds to a k-arc having cells of increasing left border. This is done during the scan. 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, and R_1 and R_2 two adjacent cells of $\mathcal{E}$ such that $R_1 \in A$, $R_1^L < R_2^L$, and R_2 adjacent to A (and thus should be added to A). If $R_2^L = R_1^R$, one just add R_2 to A, else the procedure *updates* the $k-$arc A with R_2, and may recode 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 $R_1 \cup R_2$ [15]. Then, Vacavant *et al.* consider the rectangles $R_1 - F_2$ and $R_2 - F_2$. If $R_1^R < R_2^R$, they denote $R_1 - F_2 = F_1$ and $R_2 - F_2 = F_3$. The k-arc A is finally updated with respect to five main configurations (see Fig. 3).

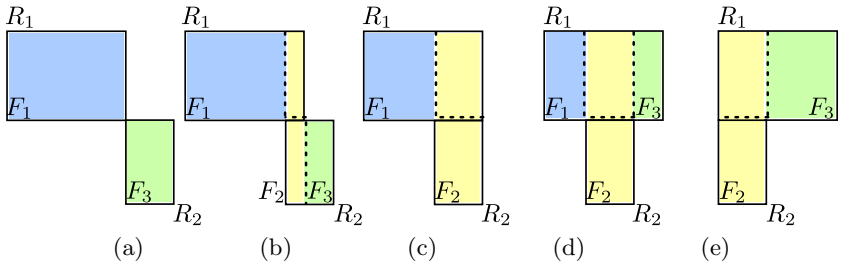


Fig. 3. Description of rectangles F_1, F_2 and F_3 in the update procedure. When $R_1^R < R_2^R$ (a and b), $R_1 - F_2 = F_1$ and $R_2 - F_2 = F_3$, else $R_1 - F_2 = \{F_1, F_3\}$ (c, d and e). If $R_1^R = R_2^L$, $F_2 = \emptyset$, when $R_1^R = R_2^R$, $F_3 = \emptyset$ and finally $F_1 = \emptyset$ in the case $R_1^L = R_2^L$.

Polygonal Reconstruction of an Irregular Object. The construction of the polygonal structure of $\mathcal{E}$ is performed by reconstructing each k-arc that recodes $\mathcal{E}$. This stage is driven thanks to a visibility cone approach inspired from [14] (see also Fig. 4). Even if this algorithm is linear-time, it is a greedy approach that could leads to some very short segments and acute angles.

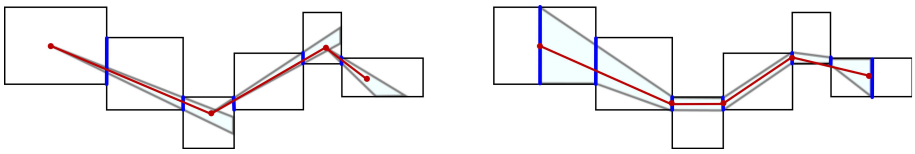


Fig. 4. The visibility cone approach incrementally produces a polygonal reconstruction with a partial preimage (left). In our contribution, we use the O'Rourke's algorithm in order to obtain a complete preimage representation (right). We also show an example of reconstruction that we describe in the section 3.

3 Unsupervised Polygonalization of Noisy Digital Contours

3.1 A Novel Approach to Vectorize Irregular Isothetic Objects

The polygonal reconstruction of $\mathcal{E}$ is again performed by reconstructing each k-arc that recodes $\mathcal{E}$. Though, instead of decomposing a given k-arc A into segments lying into a visibility cone like in [15], we decompose it into *straight k-arcs*, *i.e.* sets of k-adjacent cells that can cover a straight line and whose preimage is thus not empty. In Fig. 4 (right), we present the result of this kind of process on a simple k-arc decomposed into four straight parts.

Originally, O'Rourke's algorithm [10] aimed to solve a linear inequality system. Given n ranges $\{[\alpha_k, \omega_k]\}_{k=1,n}$ ordered by time t_k, with $t_1 < \ldots < t_n$,

this approach computes all the lines $u = mt + b$ that pass through each range, *i.e.* all pairs (m, b) such that $\alpha_k \leq mt_k + b \leq \omega_k$. In our case, the input ranges are the intersections between two successive cells of A. These intersections are vertical straight segments (possibly degenerated as a point) whose the extremity of greatest (resp. smallest) y-coordinate is called upper (resp. lower) input point. Due to the construction of the k-arc, the vertical straight segments are of increasing x-coordinate and O'Rourke's algorithm can thus be applied in order to compute the preimage of each straight part of A.

Even if O'Rourke originally explains its algorithm in the dual plane (m, b) [10], we can avoid explicit transformations and only work in the primal plane (u, t). The preimage is implicitely described by some consecutive vertices of the lower (denoted by $\mathcal{L}$) and upper (denoted by $\mathcal{U}$) convex hull of respectively the upper and lower input points.

We now describe a first algorithm that takes into account the shape of the preimage of each straight parts (Algorithm 1-C2, lines 7-25). It computes a polygonal line that completely lies within the k-arc.

In Fig. 5, we depict several iterations of this algorithm on a straight part. Points p_a and p_c are initialized as the first two points of $\mathcal{U}$, while p_b and p_d as the ones of $\mathcal{L}$ (Fig. 5-(a)). If $p_c.x > p_d.x$ (i), we move forward p_b and p_d, whereas if $p_c.x < p_d.x$ (ii), we move forward p_a and p_c. If $p_c.x = p_d.x$ (ii), we move both pairs of points. In either case, the middle of the intersection between a vertical line passing by $p_c.x$ in case (i) (Fig. 5-(c)) or $p_d.x$ in case (ii) (Fig. 5-(b)) and the preimage is the new vertex of the polygonal reconstruction. The process is linear-time and the resulting polygonal line lies inside the k-arc.

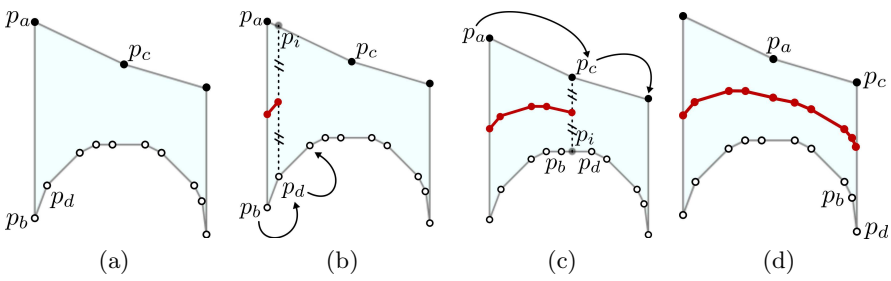


Fig. 5. Illustration of Algorithm 1-C2 on a preimage obtained from O'Rourke process. (a) is the initialization step, (b) is an update step of the reconstruction implying a lower input point, (c) a upper one, and (d) is the obtained polygonal reconstruction of the underlying straight k-arc. Algorithm 1-S2 consists in joining the first and the last point of this reconstruction.

We also develop an other algorithm (Algorithm 1-S2, lines 4-6) that constructs one straight line per k-arc passing through the middle of the first and last input range. Even if the resulting polygonal line may be partly out of the

Algorithm 1: Polygonal reconstruction of a straight k-arc based on its preimage.

input : a preimage $\mathcal{P}$ computed from a straight k-arc A, represented with its upper convex hull points $\mathcal{U}$ of size $n_{\mathcal{U}}$ and the lower convex hull points $\mathcal{L}$ of size $n_{\mathcal{L}}$, and the version selected
output: a polygonal structure computed inside $\mathcal{P}$, stored in the list of points $\mathcal{R}$

$u \leftarrow 0,\ l \leftarrow 0$;
$p_a \leftarrow \mathcal{U}[u],\ p_c \leftarrow \mathcal{U}[u+1],\ p_b \leftarrow \mathcal{L}[l],\ p_d \leftarrow \mathcal{L}[l+1]$;
$d \leftarrow p_c.x - p_d.x$;
if $version = S2$ **then** {Version S2: Simple and Straight reconstruction}
    $p_1 \leftarrow \text{middle}(\mathcal{L}[0], \mathcal{U}[0])$;
    $p_2 \leftarrow \text{middle}(\mathcal{L}[n_{\mathcal{L}} - 1], \mathcal{U}[n_{\mathcal{U}} - 1])$;
    $\mathcal{R} \leftarrow \{p_1, p_2\}$;
if $version = C2$ **then** {Version C2: Complete and Curved reconstruction}
    do
        while $d < 0 \wedge u < n_{\mathcal{U}}$ **do** {Update $\mathcal{R}$ from upper convex hull}
            Δ: line of equation $x = p_c.x$, $p_I \leftarrow \Delta \cap [p_b, p_d]$;
            $\mathcal{R} \leftarrow \mathcal{R} \cup \text{middle}(p_I, p_c)$;
            $u \leftarrow u + 1$;
            $p_a \leftarrow p_c,\ p_c \leftarrow \mathcal{U}[u]$;
            $d \leftarrow p_c.x - p_d.x$;
        while $d > 0 \wedge l < n_{\mathcal{L}}$ **do** {Update $\mathcal{R}$ from lower convex hull}
            Δ: line of equation $x = p_d.x$, $p_I \leftarrow \Delta \cap [p_a, p_c]$;
            $\mathcal{R} \leftarrow \mathcal{R} \cup \text{middle}(p_I, p_d)$;
            $l \leftarrow l + 1$;
            $p_b \leftarrow p_d,\ p_d \leftarrow \mathcal{L}[l]$;
            $d \leftarrow p_c.x - p_d.x$;
        while $d = 0 \wedge u < n_{\mathcal{U}} \wedge l < n_{\mathcal{L}}$ **do** {Update $\mathcal{R}$ from upper and lower convex hulls}
            $\mathcal{R} \leftarrow \mathcal{R} \cup \text{middle}(p_c, p_d)$;
            $u \leftarrow u + 1,\ l \leftarrow l + 1$;
            $p_a \leftarrow p_c,\ p_c \leftarrow \mathcal{U}[u],\ p_b \leftarrow p_d,\ p_d \leftarrow \mathcal{L}[l]$;
            $d \leftarrow p_c.x - p_d.x$;
    while $u < n_{\mathcal{U}} \vee l < n_{\mathcal{L}}$;
return $\mathcal{R}$;

k-object, this is an interesting way of decomposing an irregular object into a few line segments.

We show in Fig. 6 an example of the use of our contribution (computation of the complete preimage and reconstruction into line segments with Algorithm 1-C2) on an irregular object, result of a quadtree decomposition. In the following, we show how to use these algorithms to vectorize noisy digital contours.

3.2 Polygonalization of Noisy Contours by an Irregular Discrete Approach

We now propose to analyze noisy digital contour by using Kerautret and Lachaud's local noise detector [7]. This a method for estimating locally if the digital contour is damaged, what is the amount of noise and what are the highest resolution at which a part of the contour should be considered as noise-free. Depending on the selected resolution, a part of the contour is covered by a pixel of a given size at the initial resolution. The higher the amount of noise is, the biggest the pixels are.

In Fig. 7-(b), we show an example of the output of this parameter-free algorithm applied to the noisy digital object depicted in Fig. 7-(a).

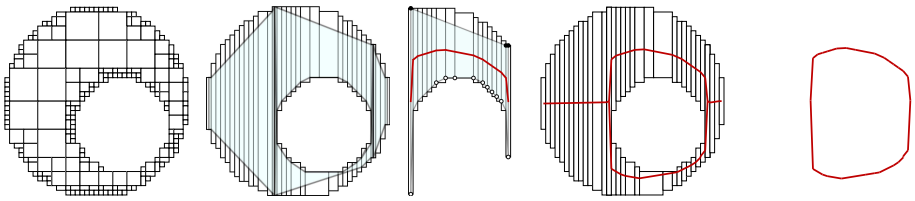


Fig. 6. Illustration of our contribution on an object digitized with a quadtree (a). (b) is the complete preimage computed on each k-arc encoding the object. One could note that the k-arc at the bottom is decomposed into two straight k-arcs. In (c), we present the reconstruction of a single k-arc, and 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 Section 3.2

As shown in Fig. 7-(b), the resized pixels overlap and thus cannot be viewed as an irregular isothetic object (Definition 1). However each resized pixel contain a given number of pixels (at the initial resolution) so that the set of resized pixels cover a subset of the input image. This subset, which is an irregular isothetic object, is the input of our reconstruction method described in the previous sections.

The input digital contour is always homotopic to a ring (one connected component and one hole). However, as in [12], the set of resized pixels may not be homotopic to the input digital contour. We can imagine that the set of resized pixels may not contain any hole or may contain more than one hole.

Thanks to the Reeb graph, which encodes the topology of the input object, we can decide whether we are in a general case (one cycle) or not (none or more than one cylce).

Moreover, in the general case, we can choose to not process the k-arcs that do not belong to the cycle so that the polygonal reconstruction is a simple polygon (Algorithm 2). For instance, only reconstructing the cycle linking nodes s, m, m, m in Fig. 7-(d) is a way of avoiding extra polygonal lines pointed by arrows in Fig. 7-(c).

Algorithm 2: Filtering of the k-arcs and the Reeb graph encoding a noisy object.

input : the set of k-arcs recoding it $\mathcal{A}$ and its associated Reeb graph $\mathcal{G}$
output: the sets $\mathcal{A}, \mathcal{G}$ are filtered in order to obtain a polygonal contour

foreach *k-arc* $a \in \mathcal{A}$ **do**
  x: the associated edge of a in $\mathcal{G}$;
  if $x = b - m \vee x = b - s \vee x = s - e \vee x = m - e$ **then**
    remove x from $\mathcal{G}$;
    remove a from $\mathcal{A}$;
return $\mathcal{A}, \mathcal{G}$;

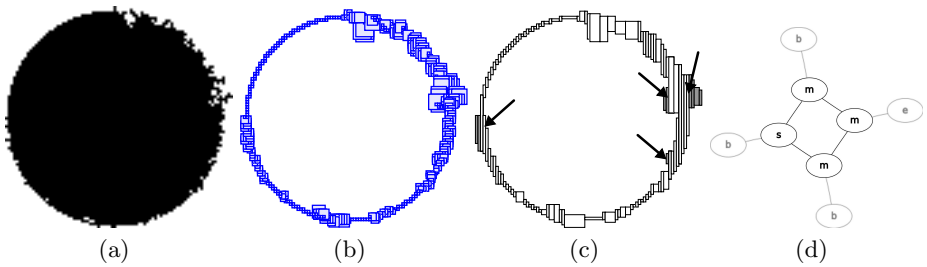


Fig. 7. From a noisy contour (a), we build a set of resized pixels (b). Then, we filter the result of our vectorization algorithm by removing k-arcs that do not belong to the polygonal minimal contour (the ones pointed by arrows). To do so, we remove their associated edges in the Reeb graph (d).

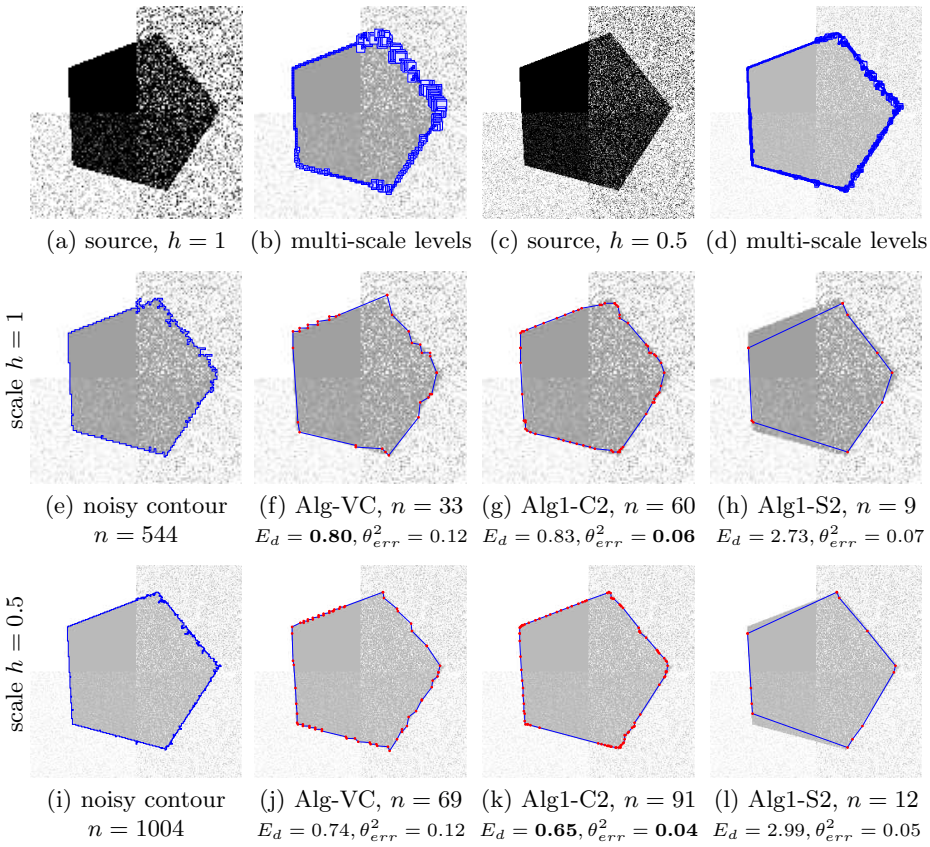
4 Experimental Results

To experiment the quality of the proposed approach, 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 (Fig. 8 (a,c)). The resized pixels (illustrated on images of Fig. 8 (b,d)) were obtained from the digital contours extracted by using a simple threshold (set to 128) (images (e,i)). The quality measures were given by 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 [8] applied on the source discrete contour.

The experiments confirm the awaited improvements provided by the Algorithm 1-C2 (Alg1-C2 in short) in comparison with the use of the algorithm based on visibility cone [15] (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 1-S2 (Alg1-S2 in short) 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.

The Algorithm 1-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 Fig. 9). We thus show that our vectorization algorithm could be applied in document analysis systems.

Finally, we compare our methods with algorithms developed by Nguyen and Rennesson [9] 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



(a) source, $h = 1$ (b) multi-scale levels (c) source, $h = 0.5$ (d) multi-scale levels

(e) noisy contour $n = 544$ (f) Alg-VC, $n = 33$ $E_d = \mathbf{0.80}, \theta^2_{err} = 0.12$ (g) Alg1-C2, $n = 60$ $E_d = 0.83, \theta^2_{err} = \mathbf{0.06}$ (h) Alg1-S2, $n = 9$ $E_d = 2.73, \theta^2_{err} = 0.07$

(i) noisy contour $n = 1004$ (j) Alg-VC, $n = 69$ $E_d = 0.74, \theta^2_{err} = 0.12$ (k) Alg1-C2, $n = 91$ $E_d = \mathbf{0.65}, \theta^2_{err} = \mathbf{0.04}$ (l) Alg1-S2, $n = 12$ $E_d = 2.99, \theta^2_{err} = 0.05$

Fig. 8. Illustration of the reconstruction algorithms. The first row shows the multi-scale levels obtained from the source contours (e,i). The second and third rows show error measures for algorithms Alg1-VC (that uses previous work), Alg1-C2 and Alg1-S2 described in this article. These results were obtained with resp. the scale $h = 1$ and 0.5

Fig. 10, 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. 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 Alg1-C2 or Alg1-S2 are comparable with the one of the NC algorithm.

5 Conclusion and Future Works

In this paper, we address the problem of reconstruction of noisy digital contours. We transform the resized pixels obtained by Kerautret and Lachaud's algorithm [7] into an irregular isothetic object recoded in a set of k-arcs whose

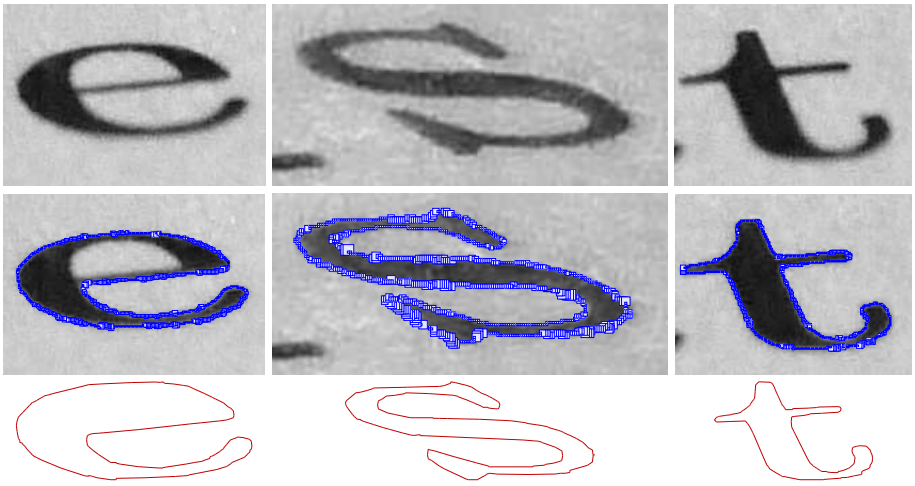
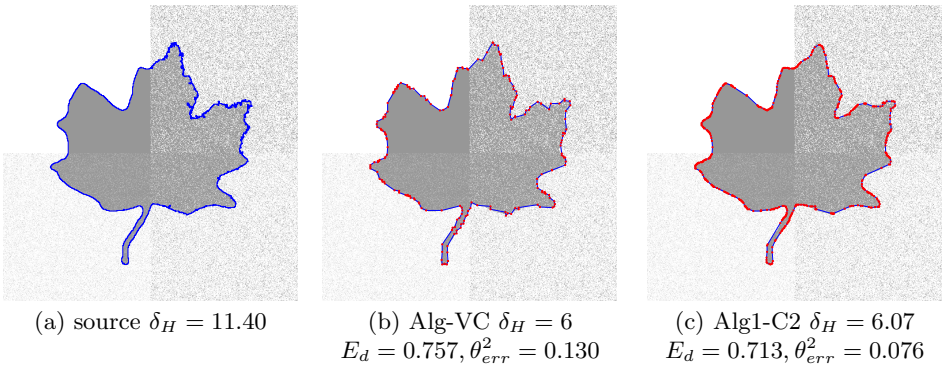


Fig. 9. The meaningful boxes extracted from scanned characters (center), and the final reconstruction we propose with Alg1-C2 (bottom).



(a) source $\delta_H = 11.40$

(b) Alg-VC $\delta_H = 6$ $E_d = 0.757, \theta^2_{err} = 0.130$

(c) Alg1-C2 $\delta_H = 6.07$ $E_d = 0.713, \theta^2_{err} = 0.076$

(d) Alg1-S2 $\delta_H = 8.92$ $E_d = 1.236, \theta^2_{err} = 0.071$

(e)Nguyen Criteria $\delta_H = 10.81$ $E_d = 1.221, \theta^2_{err} = 0.062$

(f)Marji Criteria $\delta_H = 10.63$ $E_d = 2.878, \theta^2_{err} = 0.131$

Fig. 10. Comparisons of the proposed approaches with others recent parameter free approaches [9].

topology is stored into a Reeb graph. We then vectorize it with two different linear-time methods that improve a previous work of Vacavant *et al.* [15].

As a future work, we plan to use the Reeb graph to deal with degenerate cases. We also want to consider the possibility to improve the reconstruction by using information of flat and curved parts of the processed noisy objects, since this information can be extracted from the meaningful scale detection [7].

References

1. Canny, J.: A Computational Approach To Edge Detection. *IEEE Trans. on PAMI*, **8**(6):679–698, 1986.
2. 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.
3. Coeurjolly, D. and Zerarga, L.: Supercover Model, Digital Straight Line Recognition and Curve Reconstruction on the Irregular Isothetic Grids. *C&G*, **30**(1):46–53, 2006.
4. 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.
5. Faure, A. and Feschet, F.: Linear Decomposition of Planar Shapes. In *ICPR* pp. 1096–1099, 2010.
6. Hilaire, X. and Tombre, K.: Robust and Accurate Vectorization of Line Drawings. *IEEE Trans. on PAMI*, **8**(4):890–904, 2005.
7. Kerautret, B., Lachaud, J.O.: Multi-Scale Analysis of Discrete Contours for Unsupervised Noise Detection. In *Proc of IWCIA* , LNCS 5852 pp. 187–200, 2009.
8. Lachaud, J.O., Vialard, A., and de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours, *IVC*, **25**(10):1572–1587, 2007.
9. Nguyen, T.P. and Debled-Rennesson, I.: Parameter-free method for polygonal representation of the noisy curves. In *Proc. of IWCIA*, RPS, 2009.
10. O'Rourke, J.: An on-line Algorithm for Fitting Straight Lines between Data Ranges. *Communications of the ACM*, **24**(9):574–578, 1981.
11. Reeb, G.: Sur les Points Singuliers d'une Forme de Pfaff Complétement Intégrable ou d'une Fonction Numérique. *Comptes Rendus de L'Académie ses Sciences*, Paris 222, pp. 847–849, 1946.
12. Rodriguez, M., Largeteau-Skapin, G. and Andres, E. Adaptive Pixel Resizing for Multiscale Recognition and Reconstruction. In Proc of IWCIA, LNCS 5852, pp. 252–265, 2009.
13. 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.
14. Sivignon, I., Breton, R., Dupont, F. and Andres, E.: Discrete Analytical Curve Reconstruction without Patches. *IVC*, **23**(2):191–202, 2005.
15. 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*, LNCS 4245, pp. 470–481, Szeged, Hungary, 2006.
16. 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.
17. Vacavant, A.: Fast Distance Transformation on Two-Dimensional Irregular Grids. *Pattern Recognition*, **43**(10):3348–3358, 2010.
18. Wenyin, L. and Dori, D.: From Raster to Vectors: Extracting Visual Information from Line Drawings. *Pattern Analysis and Application*, **2**(1):10–21, 1999.