Robust Decomposition of a Digital Curve into Convex and Concave Parts

Tristan Roussillon*, Laure Tougne
Laboratoire LIRIS, Université de Lyon, 5 Av Pierre-Mendès France 69676 Bron
{tristan.roussillon, laure.tougne}@liris.cnrs.fr

Isabelle Sivignon
Laboratoire LIRIS, CNRS, 8 Bd Niels Bohr 69622 Villeurbanne
isabelle.sivignon@liris.cnrs.fr

Abstract

We propose a linear in time and easy-to-implement algorithm that robustly decomposes a digital curve into convex and concave parts. This algorithm is based on classical tools in discrete and computational geometry: convex hull computation and Pick's formula.

1 Introduction

This paper focuses on the problem of decomposition of a digital curve into convex and concave parts. The main motivation is that convex and concave parts of objects coarsly determine their meaningful parts [8, 2].

Even though the concept of digital convexity has been thoroughly studied the forty past years (see [11] for a deep bibliography on the topic), such decompositions into convex and concave parts have not been studied as much. The few previously proposed methods for decomposition into convex and concave parts are either sensitive to noise [1] or approximative [4].

The main objective of the paper is to design a fast algorithm that is able to robustly perform such a decomposition. In [8], robustness is reached by applying the discrete contour evolution on the digital curve viewed as a polygonal line. This kind of structural methods are powerful for shape matching but cannot return simple indices such as the number of convex and concave parts.

We propose an original measure-based approach. A measure that decreases with deviations from convexity is assigned to a digital arc. By thresholding the measure, the arc is said convex or concave within a certain tolerance depending on the noise.

As the success of the whole process depends on the quality of the measure, we would like it to: (i) be invariant by translation, rotation and scaling; (ii) be ranging from 0 to 1, 0 for convexity; (iii) be increasing according to the amount of concavities or noise.

In section 2, we define the concept of convex and concave parts of a digital arc with a measure of digital convexity. In section 3, we show that the computation of this measure may be very efficient. An on-line algorithm that enables to robustly detect a convex or concave part is proposed in section 4. Finally, we explain in section 5 how to decompose a digital curve by iteratively running the latter algorithm. The paper ends with some conclusion and future works.

2 Digital convexity

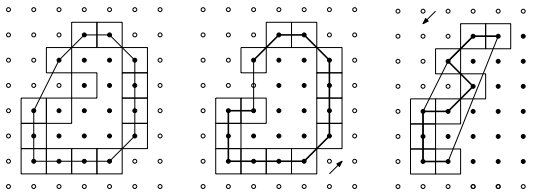
A binary image I is viewed as a subset of points of $\mathbb{Z}^2$ called digital points, located inside a rectangle of size $M \times N$. A digital object $O \in I$ is a 4-connected subset of $\mathbb{Z}^2$. Its complementary set $\bar{O} = I \backslash O$ is the so-called background. The boundary C of O, defined as the 8-connected circular list of digital points having at least one 4-neighbor in $\bar{O}$, is a digital curve. A connected subset P of C is a digital arc.

In the Euclidean plane, convexity is well defined. However several definitions of digital convexity exist (see [11] or [3]). Our definition is equivalent to the ones given, for instance, in [12]:

Definition 1 *(Fig. 1.a) Let O be a set of digital points. Let $CH(O)$ be the set of digital points located inside the Euclidean convex hull of O, $\mathcal{CH}(O)$. O is digitally convex if and only if $CH(O)$ only contains digital points belonging to O.*

Given a digital object O, the measure of digital convexity is defined as the number of points in $CH(O)$ but not in O, normalized by the size of $CH(O)$ (a missing pixel

*This author is supported by a grant from the french DGA



(a) (b) (c)

Figure 1. O (black disks) is bounded by C (squares). (a) The solid line that encloses $CH(O)$ depicts $\mathcal{CH}(O)$. (b) $P \in C$ is digitally convex. (c) $P \in C$ is neither digitally convex nor concave.

does not have the same influence on the convexity for a small and a large object).

$$convexity(O) = \frac{A(CH(O)) - A(O)}{A(CH(O))} \ . \qquad (1)$$

where function A is defined as follows:

Definition 2 *The digital area $A(O)$ of a digital object O is the number of digital points belonging to O.*

Thanks to these definitions of digital convexity (Eq. 1) and of digital area (Def. 2), it is clear that, (i) $convexity(O)$ depends on the area of the concavities, (ii) $convexity(O) = 0$ if and only if O is digitally convex and $0 < convexity(O) < 1$ otherwise, (iii) the measure is convergent while the resolution increases (since $A(O)$ and $A(CH(O))$ are convergent [7]), (iv) the measure is invariant under rigid transformations at infinite resolution and quasi-invariant otherwise.

Def. 3 is the analog of Def. 1 for digital arcs and defines convex and concave arcs.

Definition 3 *(Fig. 1.c) Let P be an oriented digital arc. The polygonal line linking all digital points of P is denoted by $\mathcal{P}$ and the shortest polygonal line linking the first and last digital point of P, such that $\mathcal{P}$ is located on its left (resp. right) side is denoted by $\mathcal{L}(P)$ (resp. $\mathcal{R}(P)$). P is digitally convex (resp. concave) if there is no digital point between $\mathcal{P}$ and $\mathcal{L}(P)$ (resp. $\mathcal{R}(P)$).*

Suppose that P is part of the boundary C of a digital object O. Moreover, suppose that P is oriented such that O is on the left of P. Let P_{left} (resp. P_{right}) be the set of digital points located between $\mathcal{P}$ and $\mathcal{L}(P)$ (resp. $\mathcal{R}(P)$). In Fig. 1.b, P is digitally convex because no digital point is between $\mathcal{P}$ (bold) and $\mathcal{L}(P)$ (superimposed to $\mathcal{P}$): $P_{left} = 0$. In Fig. 1.c, P is neither digitally convex ($P_{left} = 1$) nor concave ($P_{right} = 2$).

Eq. 2 is the analog of Eq. 1 for digital arcs.

$$\begin{array}{l} convexity(P) = A(P_{left})/A(CH(O)) \\ concavity(P) = A(P_{right})/A(CH(O)) \ . \end{array} \qquad (2)$$

Notice that the normalization by $A(CH(O))$ ensures the scale-invariance of this measure. Definition 4 relaxes this definition introducing a parameter α. We will see in section 4 that this definition leads to a robust decomposition algorithm.

Definition 4 *P is α-digitally convex (resp. concave) if $convexity(P) \leq \alpha$ (resp. $concavity(P) \leq \alpha$).*

For $\alpha = 0$, Def. 4 is equivalent to Def. 3.

3 Efficient computation of Eq. 1

The input is a digital curve C that bounds a digital object O. The algorithm runs in two steps before using Eq. 1: (i) computation of $CH(O)$, (ii) computation of $A(O)$ and $A(CH(O))$.

Since $CH(O)$ is the digitization of $\mathcal{CH}(O)$, the problem is to compute such convex hull. In classical computational geometry, the convex hull computation of n points requires a $\mathcal{O}(n \log n)$ time . However, in the digital grid, we may take profit of the intrinsec order of the digital points [5] or of its arithmetical properties [1] in order to compute such a hull in a $\mathcal{O}(n)$ time.

As C and $\mathcal{CH}(O)$ may be represented by a polygon whose vertices are digital points, Pick's formula [10] is used to efficiently compute $A(O)$ and $A(CH(O))$:

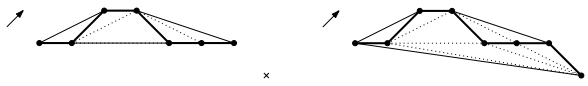
$$InAndOn(\mathcal{S}) = \mathcal{A}(\mathcal{S}) + On(\mathcal{S})/2 + 1 \ . \qquad (3)$$

$\mathcal{S}$ is a polygon the vertices of which are digital points. Functions $InAndOn()$ returns the number of digital points located on or strictly inside $\mathcal{S}$ and $On()$ returns the number of digital points located on $\mathcal{S}$. Function $\mathcal{A}(\mathcal{S})$ returns the Euclidean area of $\mathcal{S}$. In Fig. 1.a, $A_O = 14.5 + 15/2 + 1$, $A_{CH(O)} = 16.5 + 13/2 + 1$. Then, $convexity(O) = \frac{23}{24}$.

The whole process is linear in time, because computing $\mathcal{CH}(O)$ is linear in time, computing $\mathcal{A}(O)$ and $\mathcal{A}(\mathcal{CH}(O))$ as well as the number of vertices of the corresponding polygons is linear in time, and finally, applying Eq. 1 and Eq. 3 is constant in time.

4 On-line algorithm

From a digital arc P, we incrementally compute three polygonal lines: $\mathcal{P}$, $\mathcal{L}(P)$ and $\mathcal{R}(P)$. In Fig. 2, these three polygonal lines are depicted with solid lines. In the same time, we maintain $A(P_{left})$ and $A(P_{right})$, thanks to Pick's formula (Eq. 3).



(a) (b)

Figure 2. $A(P_{left})$ and $A(P_{right})$ are computed by triangulation (dotted lines) when convexity is restored.

Algorithm 1 lists operations done when a digital point is added to $\mathcal{L}(P)$. A similar algorithm may be sketched when a digital point is added to $\mathcal{R}(P)$. The data structure that enables to incrementally maintain the $\mathcal{L}(P)$ (resp. $\mathcal{R}$) is made up of a double-ended queue (deque) called leftDeque (resp. rightDeque), that has several methods running in constant time: back(), push_back(), pop_back() to respectively read, put, remove a point at the back of the deque and size() to get the number of points in the deque (like in [9]). Algorithm 1 correctly updates $\mathcal{L}(P)$ (resp. $\mathcal{R}(P)$) as well as the Euclidean area of the polygon bounded by $\mathcal{P}$ and $\mathcal{L}(P)$ (resp. $\mathcal{R}(P)$), because (i) it garantees that each triplet of consecutive vertices is counter-clockwise (resp. clockwise) oriented (ii) if not, vertices are removed from the deque and the area of the concavity is incremented until the convexity is restored (Fig. 2). Algorithm 1 is not constant at each adding, but is of order $\mathcal{O}(n)$ after n insertions, because one point is added and removed once at most. In order to compute $A(P_{left})$ and $A(P_{right})$, we apply Pick's formula (Eq. 3) on each non degenerate polygon. After a simple calculus, the last line of algorithm 1 is derived. Doing this, we assume that method size() of leftDeque (resp. rightDeque) returns all the vertices of $\mathcal{L}(P)$ (resp. $\mathcal{R}(P)$), which is not always true: all the digital points not belonging to P and located on an edge of $\mathcal{L}(P)$ or $\mathcal{R}(P)$ are not taken into account. The measure may be made exact with some computations of greatest common divisor, which increases the complexity. However, in practice, the approximated measure is used, because it is both very close and slightly less sensitive to small concavities (Fig. 3).

Algorithm 1: addPoint2leftDeque(p, n)

Input: p, end point of a digital arc P of n points
Output: $A(P_{left})$

```
1  last = leftDeque.back();
2  leftDeque.pop_back();
3  prev = leftDeque.back();
4  a = A(prev, last, p);
5  while (a < 0) do
6      leftArea += |a|;
7      last = prev;
8      leftDeque.pop_back();
9      prev = leftDeque.back();
10     a = A(prev, last, p);
11 leftDeque.push_back(p);
12 return leftArea - ((n-leftDeque.size())/2);
```

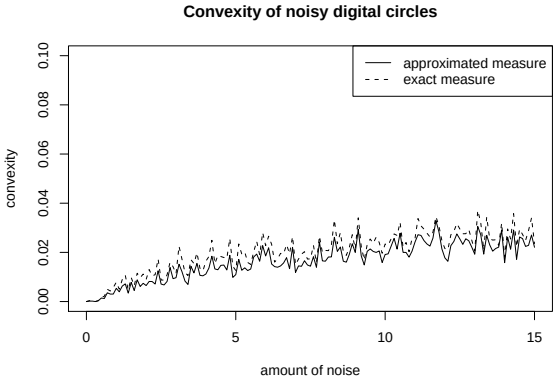
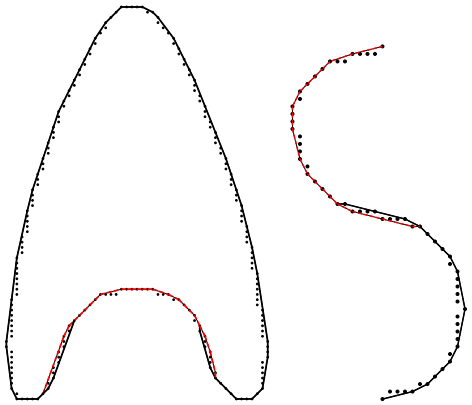


Figure 3. The approximated measure is less sensitive to noise [6] than the exact measure.

5 Shape Decomposition Process

The digital points are processed one by one along C in a counter-clockwise orientation. The core of the shape decomposition process runs in two steps: (i) the digital points that follow a starting point p, are pushed to the back of LeftDeque and RightDeque until $convexity(P) > \alpha$. Let us denote by q the last point of P, such that P is α-digitally convex. In order to make P maximal (P cannot remain digitally convex when a point is added in front or behind P), we perform the following second step: (ii) the digital points that precede p are pushed to the front of LeftDeque and RightDeque until $convexity(P) > \alpha$. When the latter expression is true, the growth of P as an α-digitally convex part stops. This algorithm is reinitialized at q and begins the detection of an α-digitally concave part P', updating $concavity(P')$ as long as an insertion to the back, then to the front of LeftDeque and RightDeque, is possible, and so on.

Since a digital point is processed twice at most, the whole decomposition is done in $\mathcal{O}(n)$, where n denotes



(a) (b)

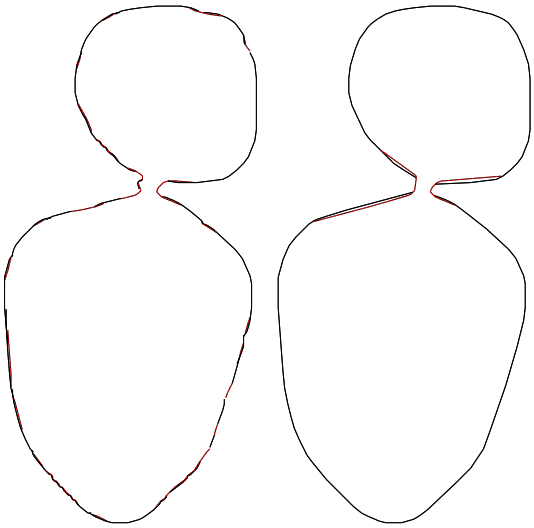
Figure 4. Letters "A" and "S" ($\alpha = 0$)

the number of digital points belonging to C.

Figures 4 and 5 illustrate this shape decomposition process respectively with synthetical digital curves and real-world images. For visualization purpose, $\forall P \mathcal{L}(P)$ and $\mathcal{R}(P)$ are drawn. For "A" [3, 2] and "S" (Fig. 4) idealized digital shapes, the natural convex and concave parts are perfectly retrieved with $\alpha = 0$. In real-world images, digital objects are often corrupted, which artificially increases the number of convex and concave parts with $\alpha = 0$ (Fig. 5.a). Slightly increasing α enables to get the expected convex and concave parts (Fig. 5.b).

6 Conclusion and Perspectives

An original, linear in time and easy-to-implement algorithm that robustly decomposes a digital curve into convex and concave parts is presented. The two main perspectives are the following: on the one hand, we think that our algorithm may be used to robustly detect digital straight line of any thickness, since a digital straight line is expected to be both digitally convex and concave, and on the other hand, we think that the multiresolution approach allowed by running our procedure for various values of α in a given range would be of great interest for shape representation.



(a) (b)

Figure 5. Convex and concave parts of a digital object that pictures two badly segmented pebbles. (a) $\alpha = 0$ (b) $\alpha = 0.01$

References

[1] I. Debled-Rennesson, J.-L. Rémy, and J. Rouyer-Degli. Detection of the discrete convexity of polyominoes. *Discrete Applied Mathematics*, 125:115–133, 2003.

[2] H. Dorksen-Reiter and I. Debled-Rennesson. Convex and concave parts of digital curves. *Geometric Properties from Incomplete Data*, 31:145–159, 2005.

[3] U. Eckhardt. Digital lines and digital convexity. In *Digital and Image Geometry*, pages 209–227, 2001.

[4] U. Eckhardt and H. Dorksen-Reiter. Polygonal representations of digital sets. *Algorithmica*, 38(1):5–23, 2004.

[5] A. Hübler, R. Klette, and K. Voss. Determination of the convex hull of a finite set of planar points within linear time. In *Elektron. Inform. Kybernet.*, pages 121–139, 1981.

[6] T. Kanungo, R. M. Haralick, H. S. Baird, W. Stuezle, and D. Madigan. A statistical, nonparametric methodology for document degradation model validation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:1209–1223, 2000.

[7] R. Klette and J. Zunic. Multigrid convergence of calculated features in image analysis. *Journal of Mathematical Imaging and Vision*, 13:173–191, 2000.

[8] L. J. Latecki and R. Lakamper. Convexity rule for shape decomposition base on discrete contour evolution. *Computer vision and image understanding*, 73(3):441–454, 1999.

[9] A. A. Melkman. On-line construction of the convex hull of simple polygon. *Information Processing Letters*, 25:11–12, 1987.

[10] G. Pick. Geometriches zur zahlenlehre. In *Zeitschrift für Natur-Wissenschafen*, pages 311–319, 1899.

[11] C. Ronse. Bibliography on digital and computational convexity. *IEEE Transactions on Pattern Analysis and Matching Intelligence*, 11(2):181–190, 1989.

[12] K. Voss. *Discrete Images, Objects and Functions in* $\mathbb{Z}^n$. Springer, 1993.