Optimal Time Computation of the Tangent of a Discrete Curve: Application to the Curvature

Fabien Feschet and Laure Tougne
{ffeschet,ltougne}@univ-lyon2.fr

Laboratoire E.R.I.C.
Université Lyon 2
5 av. Pierre Mendès-France
69676 Bron cedex, France

Abstract. With the definition of discrete lines introduced by Réveillès [REV91], there has been a wide range of research in discrete geometry and more precisely on the study of discrete lines. By the use of the linear time segment recognition algorithm of Debled and Réveillès [DR94], Vialard [VIA96a] has proposed a $O(l)$ algorithm for computing the tangent in one point of a discrete curve where l is the average length of the tangent. By applying her algorithm to n points of a discrete curve, the complexity becomes $O(n.l)$. This paper proposes a new approach for computing the tangent. It is based on a precise study of the tangent evolution along a discrete curve. The resulting algorithm has a $O(n)$ complexity and is thus optimal. Some applications in curvature computation and a tombstones contours study are also presented.

Keywords: discrete tangent, discrete curve, tombstones contours study.

1 Introduction

The work we present in this paper is inserted in a project concerning extraction of geometrical characteristics from tombstones contours. As we have to treat thousands of such figures, our goal is to find an automatic process to extract these characteristics. The most used properties of discrete curves are tangent orientation [SD91] and the curvature value in each point [VIA96b], [WS95].

In this paper, we present an optimal time algorithm for computing the characteristics of the discrete tangent in all the points of a 8-connected discrete curve. Although this algorithm is available for all the curves (open or closed) which do not self-intersect, in order to simplify the explanations we choose to present it in the case of open curves.

In section 2, we remember Vialard's algorithm [VIA96b] to compute a discrete tangent in one point of a discrete curve. This algorithm is based on Debled's algorithm [DR95] used to recognize discrete segments. The complexity of Vialard's algorithm is $O(n.l)$ where n is the number of points of the curve and l

G. Bertrand, M. Couprie, L. Perroton (Eds.): DGCI'99, LNCS 1568, pp. 31–40, 1999.

the average length of the tangent. The section 3 describes our optimal method to compute the tangent in all the points of the curve. We give the pseudo-code of the algorithm and prove its complexity. In section 4, we explain how to deduce the curvature and provide an application.

2 Vialard's algorithm

In [VIA96b] and [VIA96a], A. Vialard describes a linear time algorithm to compute the curvature in one point of a discrete curve. This algorithm proceeds in two steps: first, it computes the discrete tangent associated to the current point and second, it uses this approximation of the tangent to determine the curvature at this point. In the following, we describe the first step with precision. The second one is described in section 4.

The notion of discrete tangent is based upon the arithmetical line defined as follows.

Definition 1 ([REV91]).
A set of points $\mathcal{D} \subset \mathrm{Z}^2$ is an arithmetical line if and only if there exists four integers a, b, μ, ω such that: $\forall M = (x, y) \in \mathcal{D},\ \mu \leq ax - by < \mu + \omega$ where $\frac{a}{b}$ represents the slope of the line, μ its position in the plane and ω its thickness.

In the following, we only consider the case where ω is equal to $\sup(|\ a\ |, |\ b\ |)$. It corresponds to the usual 8-connected lines. In this paper, a curve is a set of points $\{c_0, ..., c_n\}$ (with n a natural number) such that the points c_i and c_j ($0 \leq i \leq n$, $0 \leq j \leq n$) are 8-adjacent if and only if $|j - i| = 1$ ([eAM91]).

A discrete segment is a bounded and connected subset of an arithmetical line. A discrete tangent in a point P of a curve is a subset of points surrounding P which is a discrete segment. To test if a set of points is a segment, A. Vialard uses the segment recognition algorithm of Debled and Réveillès [DR94]. For a given set of points, their algorithm finds the longest subset starting on the first point, which forms a discrete segment. Moreover, it returns the characteristics a, b and μ of the segment. The complexity of this method is $O(l)$ where l is the number of points of the recognized segment.

There exist many definitions of *the* discrete tangent in a point $P = c_i$ ($0 \leq i \leq n$) of a curve $C = \{c_0, \ldots, c_n\}$. Among those definitions, we can consider the following one.

Definition 2. *The tangent at $P = c_i$ is a segment $\{c_{i-k}, \ldots, c_i, \ldots, c_{i+k}\}$ such that $\{c_{i-k-1}, \ldots, c_{i+k+1}\}$ is not a segment (with k a natural number).*

It consists in adding pairs of points of the curve symmetrically around P until the resulting set is not a segment.
We can also try to continue the process only at one side (first on the right and second on the left) of the curve. Hence, the tangent we obtain is defined as follows.

Definition 3. *The tangent at $P = c_i$ is a segment $\{c_{i-k}, \ldots, c_i, \ldots, c_{i+k+p}\}$ (resp. $\{c_{i-k-q}, \ldots, c_i, \ldots, c_{i+k}\}$) such that neither $\{c_{i-k-1}, \ldots, c_{i+k+1}\}$ nor $\{c_{i-k}, \ldots, c_{i+k+p+1}\}$ (resp. $\{c_{i-k-q-1}, \ldots, c_{i+k}\}$) is a segment (with k and p two natural numbers).*

In definition 2, the tangent is independent of the direction of travelling along the curve and is symmetric around P. However, the tangents we obtain are smaller than the ones we have with the definition 3 and so, are less precise. A. Vialard considers the second one as we do in this article.

So, we obtain the discrete tangent in P and its characteristics a, b, μ. The complexity of this algorithm is $O(l)$ where l is the length (in number of pixels) of the tangent. As Vialard noted in [VIA96a] the execution time can be reduced by modifying the algorithm of Debled and Réveillès such that adding a point on the left is similar to adding a point on the right, thus consisting only in updates of the a, b and μ parameters. Debled's algorithm computes the characteristics of the segment with the leftmost point being the origin. However, in Vialard's algorithm, the parameters of the tangent are computed with P being the origin. Changing the origin is done in $O(1)$ by modifying the μ parameter.

Using Vialard's algorithm for each point of the curve, the global complexity of the tangent computation is $O(l.n)$ where l is the average length of the discrete tangent. When computing the tangent for all the points of a discrete segment, the complexity of the previous algorithm becomes $O(n^2)$ with n the length of the segment. This can be considered as a drawback of the algorithm. Intuitively, we expect to do the computation only on the first point and then detect all the points of the discrete segment that have a tangent with same characteristics. This is the case of the algorithm we present in the following, which complexity is $O(n)$ where n is the length of the curve.

3 A new algorithm for tangent computation

If we consider a discrete segment, the tangent is constant for all the points. So the key idea is to determine subsets of points having a constant tangent. More precisely, we look for extremal points of pieces of the curve with constant tangent. We only compute the characteristics of the tangent in these points, thus reducing the complexity of the tangent determination. In the following we describe the method with a complete example, prove its correctness and study its complexity.

3.1 The method

Let us consider an open discrete curve and a point P on this curve. By applying Vialard's algorithm in P, we compute the discrete tangent (see figure 1). We denote by R the first point we reject on the right. So, the tangent in R and in P cannot be the same ones. Moreover even if $[P, R[$ is a segment, the tangent in each of the points of this segment may have not the same characteristics. Now

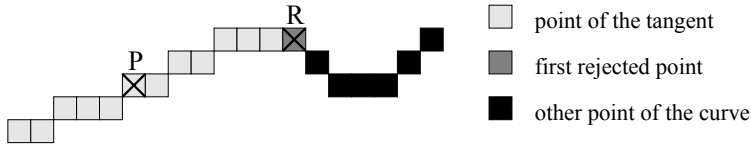

Fig. 1. The tangent in P obtained by applying Vialard's algorithm (R the first rejected point)

we look for the first point M between P and R for which the tangent has not the same characteristics as the one in P.

In figure 1, if we consider the right neighbor of P called P' and if we apply again Vialard's algorithm, we will find the same characteristics for the tangent. This comes from the fact that, in the figure, the point R can not be added to the tangent in P'. In fact, the tangent parameters remain constant for all the points for which the tangent does not contain R. M is the first point the tangent of which contains R ($\{M, \ldots, R\}$ is a segment). In order to determine the position of M, we test all the points on the left of R to find the subset of points which belong to the discrete segment starting at R (see figure 2). To do this, we make a backward read of the curve from R and we call L the first rejected point, which can be either on the left or on the right of P. Since the set of points $\{L, \ldots, R\}$ is

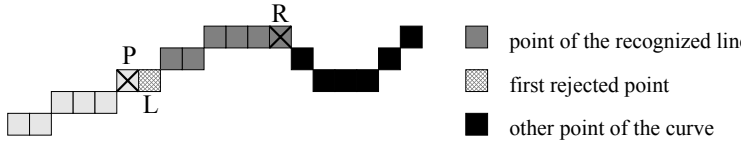

Fig. 2. Recognized line from R by applying Debled's algorithm (L the first rejected point)

not a segment but $\{M, \ldots, R\}$ is, M is a point of $]L, R]$. The goal of the following paragraphs is to prove that M is the middle of $]L, R]$.

Let us now consider the computation of the tangent in M. In a first step, we add pairs of points symmetrically around M. Then three cases can occur according to the position of M in $]L, R]$:

- L appears strictly before R in a pair. That is to say $d(L, M) < d(M, R)$. Then this pair is accepted because in this case the current computed tangent in M is included in the one in P. Hence, since the set $\{L, \ldots, R\}$ is not a segment, R does not belong to the tangent in M. So, the tangent in M has the same characteristics as the one in P. This is a contradiction with the definition of M. In conclusion, this case can not occur.
- R appears strictly before L in a pair. That is to say $d(L, M) > d(M, R)$. Then this pair is accepted because the current computed tangent in M is

included in the backward segment starting at R. Hence, L does not belong to the tangent in M which is thus different of the one in P.

– The pair (L, R) is considered. That is to say $d(L, M) = d(M, R)$. This pair is rejected. We try to add points only on one side. As we begin by the right, R is added to the current computed tangent in M. Hence, L does not belong to the tangent in M which is thus different of the one in P.

As a consequence, the characteristics of the tangent are different from the one in P for all the points closer to R than to L. That is to say for all the points P_1 that satisfy $d(L, P_1) \geq d(P_1, R)$. By definition, the point M is the first of these points. So, it is the "middle of $]L, R]$". It is defined by $d(L, M) = \lceil \frac{d(L,R)}{2} \rceil$ (see figure 3).

The characteristics of the tangent in all the points between P and M (not included) are constant. Let us note that to keep on computing the tangent in M, we can use the characteristics computed for the $]L, R]$ segment and thus, only consider points on the right of R.

To determine the tangent for all the points of the curve, we use the previous remarks. Let us notice that we do not prove yet that M is at the right of P. It is very important for the complexity of the algorithm we propose in the following section.

To prove this fact, we have to consider the following three cases. First, if L is strictly at the right of P, so M is by definition. Second, L is strictly at the left of P. So, P belongs to the segment $]L, R]$. But since R does not belong to the tangent at P, L appears strictly before R in a pair while building the tangent in P. So, by definition of M, P is strictly at the left of M. Third, if L is equal to P then, since $[L, R]$ contains at least three points, M is strictly at the right of $L = P$.

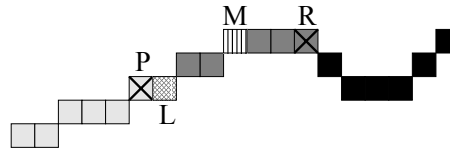

Fig. 3. Determination of the point M: the middle of the segment $]L, R]$ (the first point the tangent of which contains R)

3.2 Pseudo-code

Algorithm 1.1 Compute-tangent(curve $\mathcal{C}$)

```
1  while P belongs to C do
2      R=Tangent(P,a,b,μ)
3      if R exists then do
4          L=Reverse_DR(R)
5          M=Middle(R,L)
6          for every point I between P and M (not included) do
7              the characteristics of I are a, b, μ'
8      else
9          all points from P to the end of the curve have
10         the characteristics a, b, μ'
11         M = ∅
12     end if
13     P = M
14 end while
```

Let us now give some explanations of the previous pseudo-code. In step 2, we compute the tangent in P and, obtain the first rejected point R and the characteristics a, b and μ. R exists if we are not at the end of the curve.

The step 4 consists in recognizing the longest discrete segment from R. It returns the point L which is the first excluded point. In step 5, the middle M of the segment $]L, R]$ is determined. In lines 6 and 7, we affect the characteristics a, b and μ' to the points located between P and M (not included). μ' is computed from μ by centering the tangent on the current point [VIA96a]. With lines 9 and 10, we simply copy the characteristics a, b and μ' to all the points at the right of P. At the end, in line 13, we move to M the next point to be considered.

3.3 Efficient implementation and complexity

In this section, we propose an efficient implementation of the previous pseudo-code by modifying the step 4. In fact, we show that the `Reverse_DR` function call can be avoided by incremental computation of L.

Let us consider an iteration i of the algorithm in a point P_i as drawn in figure 4 (a). At this stage, the points L_i, R_i, M_i and G_i (which limits the tangent in P_i on the left) are known. Let us study the way we compute the next point R_{i+1} and how to determine the next point L_{i+1} without using the `Reverse_DR` function.

In figure 4 (b), the point M_i becomes the point P_{i+1}. By computing the tangent in this point with the previous algorithm, we obtain the points R_{i+1} and G_{i+1}. Let us notice that in fact, G_{i+1} is the point L_i. Indeed, since R_i and L_i are incompatible and since R_i belongs to the tangent in P_{i+1}, G_{i+1} is on the right of L_i. But L_i is the rightmost point of the curve not belonging to the discrete segment issued from R_i. Thus L_i and G_{i+1} are the same points. As a consequence, L_{i+1} is on the right of $G_{i+1} = L_i$.

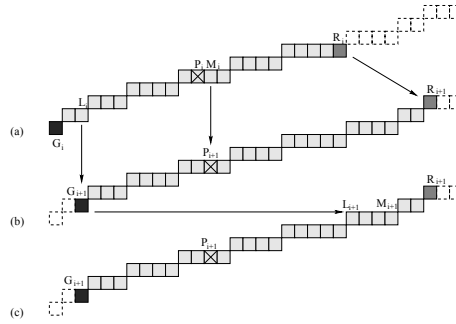

Fig. 4. Two successive computations of the tangent: the points L_{i+1} and R_{i+1} are obtained from the points L_i, R_i, M_i and G_i

Let us now determine the point L_{i+1} (see figure 4 (c)). By definition, L_{i+1} is the point that belongs to the tangent in P_{i+1} and which is the rightmost one not belonging to the discrete segment issued from R_{i+1}. So, L_{i+1} is the first point such that the set of points on its right, which belong to the tangent in P_{i+1}, plus R_{i+1} is a discrete segment. Consequently, we can proceed by iteratively deleting the leftmost point of the tangent in P_{i+1} and testing if the resulting set of points plus R_{i+1} is a segment.

Let us study the complexity of this process. Let us consider a configuration in which we have the points M_i, L_i and R_i (see case (2) of figure 5). First, adding a point on the right of R_i and checking if the resulting set is still a discrete segment (case (3)) has a complexity $O(1)$ using Debled's algorithm. Moreover, excluding a point on the left and testing if with the point R_{i+1} we always have a segment (case (4)) can also be done in $O(1)$. As a matter of fact, we have the following property ([VIA96a]).

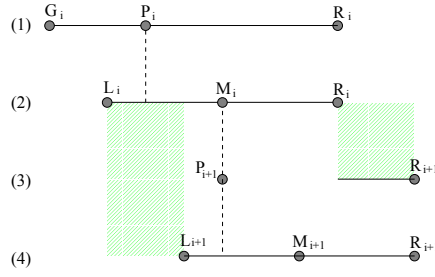

Fig. 5. Hashed rectangles cover the sets of points used during the updates of R and L

Property 1. Let $\Sigma(a, b, \mu, l)$ be a discrete segment. With no loss of generality, we suppose that we work in the first octant. Following Debled's notation [DR95],

I_0 and S_0 the principal lower and upper leaning points are also known. The characteristics $(\hat{a}, \hat{b}, \hat{\mu})$ of the segment $\Sigma \setminus (0,0)$ are obtained as follows in $O(1)$:

Algorithm 1.2

$a' = -a$, $b' = -b$, $\mu' = \mu + a * x_M - b * y_M$
$x_{temp} = x_{S_0}$, $y_{temp} = y_{S_0}$
$x_{S_0} = -x_{I_0} + x_M$, $y_{S_0} = -y_{I_0} + y_M$
$x_{I_0} = -x_{temp} + x_M$, $y_{I_0} = -y_{temp} + y_M$

if $((b' = x_{M'} - x_{S'_0})$ and $(a' = y_{M'} - y_{S'_0}))$ **then**
  $\hat{a} = y_{M'} - y_{I'_0} - 1$, $\hat{b} = x_{M'} - x_{I'_0}$
  $\hat{\mu} = (y_{M'} - y_{I'_0} - 1)x_{M'} - (x'_M - x_{I'_0})y_{M'} + 1$
  $x_{\hat{I}_0} = x_{I'_0} - b$, $y_{\hat{I}_0} = y_{I'_0} - a - 1$
  $x_{\hat{S}_0} = x_{S'_0}$, $y_{\hat{S}_0} = y_{S'_0}$
else
  if $((b' = x_{M'} - x_{I'_0})$ and $(a' = y_{M'} - y_{I'_0}))$ **then**
    $\hat{a} = y_{M'} - y_{S'_0} - 1$, $\hat{b} = x_{M'} - x_{S'_0}$
    $\hat{\mu} = (y_{M'} - y_{S'_0} - 1)x_{M'} - (x'_M - x_{S'_0})y_{M'} - b$
    $x_{\hat{S}_0} = x_{S'_0} - b$, $y_{\hat{S}_0} = y_{S'_0} - a - 1$
    $x_{\hat{I}_0} = x_{I'_0}$, $y_{\hat{I}_0} = y_{I'_0}$
  else
    $\hat{a} = a'$, $\hat{b} = b'$
    $\hat{\mu} = \mu'$
    $x_{\hat{S}_0} = x_{S'_0}$, $y_{\hat{S}_0} = y_{S'_0}$
    $x_{\hat{I}_0} = x_{I'_0}$, $y_{\hat{I}_0} = y_{I'_0}$
  end if
end if

Consequently, the treatments of each point considered in the stages (2) and (4) have a complexity in $O(1)$. Just notice, as shown in figure 5, that each point of the curve is used at most three times (including the one for storing the characteristics of the tangent between P and M). Hence, the complexity for computing the tangent in all the points of the curve is $O(n)$.

4 Some results

We use the previous algorithm to compute the curvature at each point of a discrete curve. A review of curvature estimation is given by Worring and Smeulders in [WS93]. They showed that a good curvature estimator can be obtained by a Gaussian filtering applied to the tangent orientations. More precisely, the curvature κ can be estimated by

Definition 4.

$$\kappa(P) = \frac{(\theta * \mathcal{G}'_\sigma)(P)}{1.1107}$$

with $\theta(P)$ being the angle between the tangent in P and the horizontal and with

$$\mathcal{G}_\sigma(i) = \left(\frac{1}{\sigma\sqrt{2\pi}} \exp \frac{-i^2}{2\sigma^2} \right)_{i=-3\sigma \ldots 3\sigma}$$

The coefficient 1.1107 represents the average distance between two points of the discrete curve and the window size of 6σ limits the influence of the Gaussian function.

We are involved in a project whose goal is to use classification technics and induction graphs technics to extract significant information about tombstones. These tombstones have been drawn on sheets of paper by following their contours with a pencil. Then, the resulting curves have been scanned. Moreover, we also have non geometrical characteristics such as the place where the tombstone has been found, its approximative age, its style, and so on. The goal of the project is to try to find a link between the geometrical and the non geometrical parameters. The geometry of the tombstones is represented by a set of dominant patterns. At the end, if the link exists, it would be possible to obtain an automatic classification of the tombstones as well as other information such as the movement of the population through the study of the style of the tombstones. The first part of the project concerns the computation of the dominant patterns. In order to obtain such description, we define the characteristic points as those with high curvature. To extract these points, we first proceed by smoothing and thinning the images so as to obtain strictly 8-connected curves. We apply the

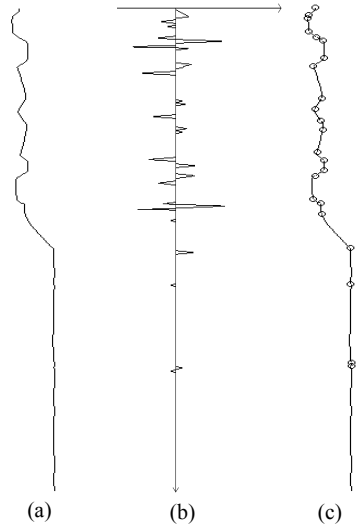

Fig. 6.(a) the tombstones contour ; (b) curvature graphs for a tombstone contour ; (c) high curvature points are marked with little circles

curvature computation on these curves. However, we have more than a thousands of tombstones contours to study and each curves is composed of two thousand or more points. Hence, we need a fast algorithm for computing the curvature. This is achieved by the linear time algorithm presented in this paper. At the end, we extract points using Teh and Chin's algorithm [TC89]. An example of curvature

computation and high curvature points on a tombstone contour is presented in figure 6.

5 Conclusion

We have presented in this article a new algorithm for the tangent computation in all the points of a discrete open curve. We also have applied this algorithm to the curvature computation on tombstones contours. Our algorithm has a complexity in $O(n)$ where n is the number of points of the curve and is thus optimal. Moreover, it is easily extensible to the case of closed curves which still do not self-intersect. This is achieved by only modifying the first step of the algorithm. More precisely, we apply once Vialard's algorithm for the first point and then use our algorithm. The complexity remains the same one.

In future works, we would like to study what happens with not strictly 8-connected curves or in a more general case, with thick lines.

Acknowledgment

The authors thank the Pr. S. Miguet for his meticulous proofreading of an earlier version of the manuscript as well as for his constant support.

References

DR94. I. DEBLED and J.P. REVEILLES. A linear algorithm for segmentation of digital curves. In *Third International Workshop on parallel Image Analysis*, June 1994.

DR95. I. DEBLED-RENNESSON. *Étude et reconnaissance de droites et de plans discrets*. PhD thesis, Université Louis Pasteur, 1995.

eAM91. Jean-Marc Chassery et Annick Montanvert. *Géométrie discrète en analyse d'images*. Editions Hermès, 1991.

REV91. J.P. REVEILLES. *Géométrie discrète, calcul en nombres entiers et algorithmique*. PhD thesis, Université Louis Pasteur, 1991.

SD91. A. W. M. SMEULDERS and L. DORST. Decomposition of discrete curves into piecewise straight segments in linear time. *Contemporary Mathematics*, 119:169–195, 1991.

TC89. CHO-HUAK TEH and ROLAND T. CHIN. On the detection of dominant points on digital curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):859–872, August 1989.

VIA96a. ANNE VIALARD. *Chemins Euclidiens : un modèle de représentation des contours discrets*. PhD thesis, Université Bordeaux I, 1996.

VIA96b. ANNE VIALARD. Geometrical parameters extraction from discrete paths. In *6th International workshop DGCI*. Springer, 1996.

WS93. MARCEL WORRING and ARNOLD W.M. SMEULDERS. Digital curve estimation. *CVGIP: Image Understanding*, 58(3):366–382, 1993.

WS95. MARCEL WORRING and ARNOLD W.M. SMEULDERS. Digitized circular arcs: Characterization and parameter estimation. *IEEE PAMI*, 17(6):587–598, jun 1995.