On Recursive, $O(N)$ Partitioning of a Digitized Curve into Digital Straight Segments

M. Lindenbaum and A. Bruckstein

Abstract— **A simple on-line algorithm for partitioning of a digital curve into digital straight-line segments of maximal length is given. The algorithm requires $O(N)$ time and $O(1)$ space and is therefore optimal. Efficient representations of the digital segments are obtained as byproducts. The algorithm also solves a number-theoretical problem concerning nonhomogeneus spectra of numbers.**

Index Terms— **Chain code, digital straight lines, on-line algorithms, number theory.**

I. Introduction

When a straight edge is digitized on a square grid, we obtain a sequence of grid points defining a digital straight-line segment. The characterization of such sequences and their application has drawn considerable attention (see [1]–[14] for a partial list of publications and [15] for a recent bibliography of this subject).

Digital straight lines were used for length and slope estimation from digitized images [6], for achieving sub-pixel accuracy [7] and sub-pixel registration [8], and for coding of line drawings [9]. An initial preprocessing stage of partitioning an arbitrary digitized edge into subsequences that are digital straight lines of maximum length is required in these applications.

To partition a digital curve into subsequences that are digital straight segments, one must have an efficient test for digital straightness, i.e., a method to decide whether a given sequence of n grid points is a digital straight segment. Several tests for straightness were proposed. The first rigorous test was the chord property, which was derived by Rosenfeld [1]. Some others were the convex hull method proposed by Kim and Rosenfeld [2], the hierarchical decomposition of straight lines suggested by Wu [3], and linear programming suggested by Werman *et al* [10]. The latter three algorithms require $O(N)$ time and are optimal. Checking whether a given sequence of grid points is a digital straight segment is equivalent to the problem of finding a line that passes through a given set of vertical segments for which an $O(n)$ optimal dynamic algorithm was proposed by O'Rourke [26].

It is interesting to note that an equivalent problem was formulated as a number-theoretical subject via the notion of spectra of numbers. A sequence $\{a_i\}_{i=1}^{n}$ is called a nonhomogeneous spectrum if there exist two numbers α and β s.t. $a_i = \lfloor \alpha i + \beta \rfloor$, where $\lfloor x \rfloor$ is the largest integer not exceeding x. When $\beta = 0$, we speak about homogeneous spectrum. The characterization and construction of homogeneous spectra was investigated by Bernoulli and Markoff [16], Stolarsky [17], Fraenkel *et al.* [18], and Graham *et al.* [19]. Boshernitzan and Fraenkel [20] characterized nonhomogeneous spectra and suggested an $O(N)$ algorithm for checking whether a given sequence is a nonhomogeneous spectrum. It is easy to recognize that the question whether a given sequence is a (nonhomogeneous) spectrum and the question whether a given sequence of grid points is a digital straight line are identical. (See more about the relation between digital lines and some classic problems in mathematics in the discussion of [13].)

Partitioning a digital curve into straight segments may be recursively performed as follows. Starting from an endpoint of the digitized curve, a subsequence of grid points is tested for straightness. Then, the next grid point is added to it, and the new subsequence is again tested for straightness. The process is repeated until a subsequence of length $L+1$ fails the test. Then, the subsequence of L points is a digital straight segment, and at the next point on the digitized curve, tests for the next segment are restarted. If, in the described algorithm, one of the methods for testing straightness is applied, then detecting a straight segment of length L requires $\sum_{k=2}^{L+1} 0(k) = 0(L^2)$ computations, and the complete decomposition of a curve of length N into straight segments takes $O(N^2)$ time.

Recently, a new algorithm capable of decomposing a digital curve into digital straight segments in $O(N)$ time was proposed by Smeulders and Dorst [21]. This algorithm, which gives a major improvement over most methods, is rather complicated to understand because it is based on the exploitation of the hierarchical structure of digitized lines [3], [5], [13]. This correspondence proposes an alternative algorithm for curve decomposition, which has similar performance, and is much simpler to understand and to implement. In the following section, we describe the principle underlying the algorithm and follow that with a formal representation of the basic two-pass algorithm. Then, we show how this algorithm may easily be modified into a one-pass version. In the following section, we show that two alternative, efficient representations of the detected line segments can be found as byproducts of the algorithm, and we discuss how these representation may be recursively updated. We conclude with a short discussion that compares our new algorithm with the previous ones and show its relation to the theoretical question of spectra.

II. Principles

Consider a straight line $y = m_o x + b_o (0 \le m_o, b_o \ge 1)$ digitized on an $N \times N$ unity grid into the digitized line segment

$$L_o \triangleq L(m_o, b_o, , N) \triangleq \left\{ (x_i, y_i) \middle| \begin{array}{c} x_i = i \\ y_i = \lfloor m_o x_i + b_o \rfloor . \\ i = 0, 1, 2, \ldots N \end{array} \right\}$$

Instead of describing the digital line by a list of its grid points, one can specify the left-most grid point ($x_i = 0$) followed by a Freeman chain code that describes the transitions between adjacent digital line points. A typical property of digital lines is that only two types of chain code elements (links) appear in this chain code and that one of them always appears in groups (runs), and the second always appears isolated. These links are called the majority link and the minority link, respectively. When the generating line is in the first quadrant, as specified above, the chain code elements c_i are equal to the differences $c_i = y_{i+1} - y_i$, which take only 0 and 1 values, and a typical digital line chain code is 01101110110.

Define the preimage of the digital line segment L_o as the set of all lines $y = mx + b$ having the same digitized form L_o. A sufficient and necessary condition for a line $y = mx + b$ to be in the preimage of L_o is clearly

$$\lfloor m_o i + b_o \rfloor \le mi + b < \lfloor m_o i + b_o \rfloor + 1 \quad 0 \le i \le N.$$

Let (m, b) denote the plane of parameters of straight lines. Let $D(L_o)$ be the domain of parameters in this plane, which includes

Manuscript received July 29, 1991; revised June 20, 1992. Recommended for acceptance for Associate Editor D. P. Huttenlocher.

The authors are with the Department of Computer Science, Technion—Israel Institute of Technology, Haifa, Israel.

IEEE Log Number 9209992.

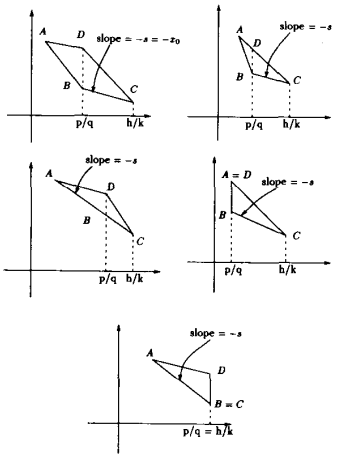

Fig. 1. Five kinds of domains that are possible in the parameter plane.

the parameter of lines in the preimage of L_o. Each point in this domain satisfies

$$\left.\begin{array}{l} b \geq -mi + \lfloor m_o i + b_o \rfloor \\ b < -mi + \lfloor m_o i + b_o \rfloor + 1 \end{array}\right\} 0 \leq i \leq N.$$

Thus, the domain $D(L_o)$ is the intersection of $2N + 2$ half planes or $N + 1$ stripes of height 1.

$$D(L_o) = \cap_{i=0}^{N} \left\{ (m,b) \;\middle|\; \begin{array}{l} b \geq -mi + \lfloor m_0 i + b_0 \rfloor \\ b < -mi + \lfloor m_0 i + b_0 \rfloor + 1 \end{array} \right\}$$

The (m,b) plane $(0 \leq m, b \geq 1)$ is partitioned into such domains, and each contains the parameters of a different preimage. The following interesting and important properties of these domains were derived by Dorst and Smeulders [4] and by McIlroy [11].

Property 1: *Each of the domains is a convex polygon with at most four vertices. If the domain has four vertices, then two of them have a common m coordinate, which is between the m coordinates of the other two vertices.*

Each of the lines that partition the parameter plane into domains has an integer slope x_o. For a given length N of the digital line, the partition of the parameter planes into domains satisfies the following interesting property.

Property 2: *The vertices that lie on each of the separating lines have m coordinates that form a Farey series of order $max(x_o, N - x_o)$.*

The Farey series of order n, which is denoted by $\mathcal{F}_n$, is defined as the ordered series of the fractions between 0 and 1 whose denominator is not bigger than n; see [22]. Thus, for example

$$FF_6 = \left\{0, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{5}{6}\right\}.$$

It was also shown that the domains may take only one of five possible characteristic shapes, which are illustrated in Fig. 1. Note that in the four-sided domain, two vertices B and D have a common m coordinate. We use the following uniform notation for all domains. The upper left vertex is called A, the lower right vertex is called C, and the intermediate vertices are called B and D. In the case of three-sided domains, a "generalized" vertex $B(D)$ is added with the same m coordinate as $D(B)$. In some of the cases when the region has a side that is parallel to the b axis, two vertices may coincide (see Fig. 1).

Consider a set of N grid points

$$\{(x_i, y_i) | x_i = i \;\; i = 0, 1, ..., N\}$$

which correspond to the nonempty domain in the (m,b) plane

$$D_N = \cap_{i=0}^{N} \{(m,b) | -mx_i + y_i \leq b < mx_i + y_i + 1\}$$

and hence constitute a digital straight-line segment of length N.

Suppose an additional grid point $(x_N + 1 = N + 1, y_N + 1)$ is given. The domain D_N may or may not be crossed by the new separating lines related to the new grid point (and thus may or may not be split). However, if it is crossed by the line, then the following interesting properties hold.

Property 3a: *Only one of the two lines related to the new grid point crosses the domain D_N.*

Property 3b: *The separating line passes through one (and only one) of the points A, B, C, D.*

Property 3c: *A new vertex, which is created on one of the sides, say, AB, with slope x_{AB}, has an m coordinate that has the following properties. It is a rational number with denominator $N+1-x_{AB}$ and is also the member of the Farey series of order $N + 1 - x_{AB}$, which succeeds the m coordinate of A and precedes the m coordinate of B.*

Checking whether the set $\{(x_i, y_i) | i = 1, ...N + 1\}$ is a digital line is equivalent to checking whether the set

$$D_{N+1} = D_N \cap_{i=0}^{N} \left\{ (m,b) \;\middle|\; \begin{array}{l} b \geq -mx_{N+1} + y_{N+1} \\ b < -mx_{N+1} + y_{N+1} + 1 \end{array} \right\}$$

is nonempty. From Property 1, and as argued in the next section, this intersection may be done in constant time and hence serves as the basis of an $O(N)$ *dynamic, on-line* test for straightness that underlies the algorithm presented in the next section.

III. An $O(N)$ Algorithm for Decomposing a Digital Curve into Maximal Length Digital Straight-Line Segments

We start by presenting a two-pass basic algorithm for decomposing a digital curve into digital straight segments. This algorithm follows directly from the principles described in the previous section, and we feel it is easier to understand. Then, at the end of this section, we show how to transform it into a one-pass, on-line algorithm. In the next section, we shall show an interesting variation, which depends on the special properties of the parameters' domains, to the principal step of the the algorithm.

The domains in the parameter plane, which are described in the previous section, are included in $\{(m,b) | 0 \leq m, b \leq 1\}$. Hence, the usual chain code [3] of the corresponding digital line segments includes only the "0" and "1" links. The use of the (m,b) domains for finding straight line segments in the general digital curve must be preceded by a "normalization" stage in which all links are transformed into "0" or "1" links. Thus, the algorithm starts by identifying the general direction of the curve (phase 1), the two types of links that are part of the first segment are found and denoted majority ("0") and minority ("1") links. Then, the longest digital straight segment from the beginning of the curve is found and deleted from the curve (phase 2). The process is repeated until the entire curve is decomposed into straight segments.

Assume the curve is given by a starting point and an eight-connected chain code $C_1, C_2, ..., C_N$. Let majority type and minority type be two variables, which are initially undefined, that may take any two consecutive links as values.

A. The Algorithm

Phase 1: Start from the first link C_1 and proceed along the curve. Stop at the first link (the kth one in the order of checking) that satisfies one of the following conditions:

1) If the series of the first k links contains two adjacent links of the same type (e.g., $C_{j-1} = C_j = 3$), then denote this type the majority type, and continue in phase 2.
2) If the series of the first k links contains two types of links that are not adjacent (e.g., $C_i = 3$ and $C_j = 5$) or if the curve terminates at the kth link, denote one of the types of the first $k-1$ links (the majority link), and continue in phase 2.

Phase 2:

a) (Initialization)

1. Let $P \doteq (P_x, P_y) = (0,0)$.
2. Let $D = D_0 = \{(m,,b) | 0 \le m, b \le 1\}$.

Repeat steps b)–f) for $i = 1, 2$.

b) (Changing the original chain code into a new one, which comprises only "1-s and "0-s)

- If the ith link C_i is of the majority type, replace it with $C_i = 0$.
- If the minority type is not yet defined, and the link C_i is different from the majority type but is adjacent to it, define its type as the minority type.
- If the ith link is of the minority type, replace it with $C_i = 1$.
- If the ith link is neither of the majority type nor of the minority type, then it does not belong to the current digital segment. Let $L = i - 1$, and continue to step g.

c) (Finding the next point on the normalized grid) $\overline{P} \doteq \overline{P} \dot{+} (1, C_i)$.

d) (Finding the new parameter domain) $D = D_o \cap \{(m,,b) | -m \cdot P_x + P_y \le b < -m \cdot P_x + P_y + 1\}$.

e) (Is the new domain nonempty?) If $D = \emptyset$, then let $D = D_o, L = L - 1$, and continue in step g. If $D \neq \emptyset$, then let $D_0 = D$.

f) If C_i is the last link (the curve terminates), let $L = i$.

g) The first L links comprise a digital line segment characterized by the domain D. Delete these links from the digital curve, and continue in phase 1.

A digital straight line may include at most two types of links that must be adjacent. This feature is implicitly used in the algorithm. Another important feature of digital lines is that the domain of the preimage in the parameter plane must be in a convex polygon with no more than four sides. This implies that the intersection between D_o and the infinite strip done in step d) of phase 2 may be implemented by any reasonable intersection algorithm in a constant time (see, for example, the polygon intersection algorithm on pp. 263–269 of [23]). Since for a segment of length L the steps a)–f) are performed at most $L+1$ times each, and each step is performed in constant time, it follows clearly that the full curve of length N is partitioned into digital straight segments in $0(N)$ time.

Note that each of the lines separating the regions in the (m,b) plane may be described as $b = -mx_o + y_o$, where x_o and y_o are integers smaller than N. It follows that intersecting the polygonal regions as well as representing them may be done using only integer arithmetics, thus preventing any rounding errors. In the next section, we derive a variation to the region updating step, which eliminates the need to use any intersection algorithm.

A speedup of the algorithm may be obtained by observing that only grid points in the beginning and end of the majority link runs limit the parameter domain. Hence, the intersection of domains may be performed only for these points.

Consider the following modification of the algorithm presented above. The first phase is totally ignored. Each step in phase 2 is done twice. It is done once under the assumption that the first link in the subsequence is the majority link, and the other type of link is the minority link (if, of course, it is adjacent to the first type). Then, the step is repeated for the opposite assumption: that the first link is the minority link. (Of course, the number of variables is doubled since one must hold all variables for both assumptions.) For most cases, after a small number of points, one of the links appears in a run of two links or more, and one of the assumptions is confirmed, and thus, the algorithm may proceed with only the correct option. The on-line algorithm created this way considers only one point at a time and does a constant number of operations for it. Furthermore, it does not need to store the input and output sequences and, hence, requires only $O(1)$ space for storing the varying parameter space region.

IV. Finding Two Representations of the Digital Line Segments

Sometimes, it is required that a description of the digital segment that is more concise and informative than a list of the L links be found.

One such description was introduced by Dorst and Smeulders in [4]. They proved that any digital segment can be characterized uniquely by a quadruple of integers (n, q, p, s), where n is the length of the segment, and q is the period of the sequence of the n links. If a subsequence of length q is repeated periodically, an infinite digital line is created in which the given digital segment is included. The preimage of this infinite line includes only lines whose slope is p/q. The parameter s, which may take integer nonnegative values smaller than q, differentiates between digital lines with similar first three parameters and has a meaning of phase. This description may be easily found for each digital segment found by the partition of the curve. Dorst and Smeulders have shown that the vertices of the domain of parameters of the preimage have at most three distinct values of m coordinates (i.e., if the domain has four vertices, two of them have identical m coordinates) and that p/q is the intermediate m coordinate (see Fig. 1). Hence, by choosing the (unique) intermediate value out of the m coordinates of the domain vertices, p/q are immediately obtained. n is simply the length of the segment, and s may be found by the following argument. The line with slope p/q and minimal b, which is still digitized into the digital line segment, cross a grid point at $x_i = s$ (see [4]) and is mapped into the point B in the parameter plane (see Fig. 1). The line with the maximal slope that is still digitized into the digital line segment also crosses the same point and is mapped into the point C in the parameter plane. The point $x_i = s$ itself is mapped into the line BC, which has the slope s, and thus, if the region is known, then so is the parameter s.

Another parameterization was suggested by Lindenbaum and Koplowitz [12]. They suggest the use of a different quadruple of integers (n, k, h, x_o) defined indirectly from the digital line. n is the length of the digital line, h/k is the maximal m coordinate in the parameter domain, and x_o has, again, the meaning of phase. Although this description lacks the intuitive appeal of the Dorst and Smeulders one, it has one advantage. The domain of quadruple integers that are parameters of straight lines is given explicitly and simply. Thus, it is more economic for storage and encoding. The parameters h, k are found from h/k, which is the m coordinate of the rightmost vertex, n is simply the length of the digital segment, and x_o (which is identical to the parameter s in the Dorst and Smeulders parameterization) may be found in the same way (see Fig. 1).

It is interesting to note that using the special characteristics of the parameters domains, we can substitute the region intersection step

by a simpler one. Recall that when the domain is changed, the new separating line must cross one of its vertices. Let the two lines related to the new curve point (x_{N+1}, y_{N+1}) be

$$L_{\text{low}}^{N+1} = \{(m,b)|b = -mx_{N+1} + y_{N+1}\}$$
$$L_{\text{high}}^{N+1} = \{(m,b)|b = -mx_{N+1} + y_{N+1} + 1\}$$

and let the region between them be

$$R^{N+1} = \{(m,b)| - mx_{N+1} + y_{N+1} \leq blt - mx_{N+1} + y_{N+1} + 1\}.$$

The domain D_N is represented, for convenience, by all vertices and all sides. (We could, of course, use one of the efficient representations discussed above, but that would lead to increased complication and to higher time complexity.)

Each vertex is represented by two (rational) coordinates, where each is stored as two integers $(A = (m_A, b_A))$, and each side is represented by two (integer) parameters of the line $(AB = (x_{AB}, y_{AB}))$. Let $z_n^+(x)$ be the member of $\mathcal{F}_n$ that succeeds x. Let $z_n^-(x)$ be the member of $\mathcal{F}_n$ that precedes x.

The region intersection step can be substituted by the following rule, which follows directly from Property 3.

Updating Rule:

1) If $B \cap L_{low} \neq \emptyset$,
   then (A and AB are changed.)
   $m_A = z^+_{N+1-x_AD}(m_A)$, $b_A = -m_A x_{AD} + y_{AD}$, $x_{AB} = x_{N+1} = N+1$, $y_{AB} = y_{N+1}$.
2) If $D \cap L_{low} \neq \emptyset$,
   then (A, B, D and AB, AD are changed.)
   $m_A = m_D$, $b_A = b_D$, $m_B = m_D = z^-_{N+1-x_{BC}}(m_C)$, $b_B = -m_B x_{BC} + y_{BC}$, $b_D = -m_D x_{CD} + y_{CD}$, $x_{AB} = N+1$, $y_{AB} = y_{N+1}$, $x_{AD} = x_{CD}$, $y_{AD} = y_{CD}$.
3) If $B \cap L_{high} \neq \emptyset$,
   then (B, C, D and BC, CD are changed.)
   $m_C = m_B$, $b_C = b_B$, $m_B = m_D = z^+_{N+1-x_AD}(m_A)$, $b_B = -m_B x_{AB} + y_{AB}$, $b_D = -m_D x_{AD} + y_{AD}$, $x_{CD} = N+1$, $y_{CD} = y_{N+1} + 1$, $x_{BC} = x_{AB}$, $y_{BC} = y_{AB}$.
4) If $D \cap L_{high} \neq \emptyset$,
   then (C and CD are changed.)
   $m_C = z^-_{N+1-x_{BC}}(m_C)$, $b_C = -m_C x_{BC} + y_{BC}$, $x_{CD} = x_{N+1} = N+1$, $y_{CD} = y_{N+1} + 1$.
5) If $B \in R^{N+1}$ and $D \in R^{N+1}$,
   then (no change: $D_{N+1} = D_N$).
6) Else $D_{N+1} = \emptyset$.

All operations in the above updating of the domain are justified directly by Property 3. Consider, for example, the domain described in Fig. 2. If the line L_{low} passes through the point D, then the region is changed into the triangular region $D_{N+1} = A'B'C'$. (This is the second case in the IF sentence.) The new vertex A is identical to the old vertex D. The new vertex C is identical to the old vertex C. The new vertex B must be the member of $\mathcal{F}_{\max(N+1-x_{BC}, x_{BC})}$. If $\max(N+1-x_{BC}, x_{BC}) = x_{BC}$, then the point B' would be vertex of some domain when only N points are considered. Since this is not the case, it follows that $max(N+1-x_{BC}, x_{BC}) = N+1-x_{BC}$.

Finding successive members in some Farey series would require $O(\log N)$ computation since it is needed to solve an integer equation that comes out of a known property of any Farey series in which two successive members $\frac{a}{b}$ and $\frac{c}{d}$ ($\frac{a}{b} < \frac{c}{d}$) satisfy the relation

$$bc - ad = 1$$

[22], [24]. In our case, the situation is more fortunate. Since we know that only members with a denominator equal to the new order of the series are added to existing members, it follows that the successive number of the Farey series may be found in constant time.

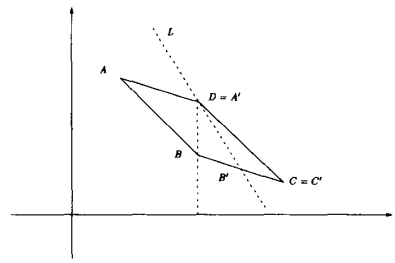

Fig. 2. Example of the partition of a domain when a new point is added.

Since $\frac{p}{q} = m_B$, $\frac{h}{k} = m_C$, and $s = x_o = x_{BC}$, it follows that the above procedure is a straightforward method for recursively updating each of the digital line representations.

If the digital straight segment is long, then encoding it using one of these representations can be very efficient since encoding four parameters requires $O(\log N)$ bits, in comparison with $O(N)$ bits required for direct storage of the chain code. This observation is the basis of the curve-encoding method proposed in [9]. There, a digital curve is partitioned into a sequence of digital straight segments using a tree state diagram. Alternatively, one could use the proposed recursive partition algorithm and save the space required to store the diagram.

V. Discussion

We have presented a simple and efficient algorithm for partitioning a digital curve into digital straight segments. This algorithm has an on-line nature and requires a constant number of operations for adding each new point. The total number of operations required for partitioning a digital curve of length N is $O(N)$, and the maximal space required is $O(1)$. The algorithm facilitates the classical applications of digital lines, such as estimating the length of digitized curves [25], subpixel reconstruction [7] and registration [8], and boundary coding [9].

The algorithm for finding a line that passes through a given set of line segments, which was proposed by O'Rourke [26], is very similar and could be used here. It is also a dynamic algorithm with $O(N)$ time complexity. However, since it does not rely on digital line properties, the number of operations done for each additional point may be as large as $O(N)$, and the space it requires is also $O(N)$.

The algorithm proposed by Smeulders and Dorst [21] takes a linguistic approach and checks the linearity of the digital curve segment by dynamically decomposing it into a hierarchy of "runs." This method is relatively complicated to understand and to prove, relies on four kinds of runs that are kept for each level in the hierarchy, and are updated with the inclusion of each additional point. The linearity condition is then checked for all levels. Fortunately, adding a point to a digital line usually affects only the lower levels of the hierarchy, and thus, only these levels are updated and checked for linearity. It follows that on the average, the algorithm proposed in [21] needs only $O(N)$ simple computations.

Our geometric approach yields an alternative algorithm that is easier to understand and to implement. Each point on the digital line implies two constraints on the parameters of the underlying line in the form of two simple linear inequalities. It is straightforward to demand that if a string of points is really a digital line, then all these constraints must agree, and their intersection region in the parameter space is nonempty. It is the special chracteristics of this intersection region, which we exploit, that make its computation particularly easy and efficient. The range intersection operation is substituted into an

update of a small constant number of integer variables and thus allows a very simple implementation of the digital linearity criterion.

All three algorithms take $O(N)$ steps to decompose a digital curve into digital straight segments. The proposed algorithm uses only a constant $O(1)$ storage and is the only one that has on-line characteristics and requires $O(1)$ steps for each additional point. In these respects, it is advantageous compared with the O'Rourke and Smeulders and Dorst algorithms, which use $O(N)$ and $O(\log N)$ storage, respectively, and may require up to $O(N)$ and $O(\log N)$ steps for some points and cannot be used as on-line algorithms. The Smeulders and Dorst algorithm requires only additions and state changes to compute the representation and for checking the linearity condition, and thus, it is possible that it will turn out to be faster on some machines than the proposed algorithm, which sometimes requires integer multiplications.

As shown previously, the problem of finding whether there is a line $y = mx + b$ that is digitized into a given digital line is identical to the problem of finding whether there are two numbers m and b satisfying $\lfloor mi + b \rfloor = y_i$, where $\{y_i\}_{i=1}^{N}$ is the given y coordinate of the digital curve. Substituting m for α, b for β, and y_i for a_i, it is clear that this problem is identical to the problem of finding whether a given sequence $\{a_i\}_{i=1}^{n}$ is a nonhomogeneous spectrum. Hence, we also have a new algorithm for determining nonhomogeneous spectra, which is simple, dynamic, efficient, and requires very little space. A simple modification (of limiting the parameters pairs to the $b = 0$ axis) changes the proposed algorithm into an algorithm for determining homogeneous spectra.

References

[1] A. Rosenfeld, "Digital straight line segments," *IEEE Trans. Comput.*, vol. C-23, pp. 1264–1269, 1974.

[2] C. E. Kim and A. Rosenfeld, "Digital straight lines and convexity of digital regions," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-4, pp. 149–153, 1982.

[3] L. D. Wu, "On the chain code of a line," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-4, pp. 347–353, 1982.

[4] L. Dorst and A. W. M. Smeulders, "Discrete representation of straight lines," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-6, pp. 450–463, 1984.

[5] S. H. Y. Hung, "On the straightness of digital arcs," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-7, pp. 203–215, 1985.

[6] L. Dorst and A. W. M. Smeulders, "Best linear unbiased estimators for properties of digitized straight lines," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-8, pp. 276–282, 1986.

[7] J. Koplowitz and A. P. Sundar Raj, "A robust filtering algorithm for subpixel reconstruction of chain coded line drawings," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-9, pp. 451–457, 1987.

[8] C. A. Berenstein, L. N. Kanal, D. Lavine, and E. Olson, "A geometric approach to subpixel registration accuracy," *Comput. Vision Graphics Image Processing*, vol. 40, no. 3, pp. 334–360, 1987.

[9] M. Lindenbaum and J. Koplowitz, "Compression of chain codes using digital straight line sequences," *Patt. Recogn. Lett.*, vol. 7, pp. 167–171, 1988.

[10] M. Werman, A. Y. Wu, and R.A. Melter, "Recognition and characterization of digitized curves," *Patt. Recogn. Lett.*, vol. 5, pp. 207–213, 1987.

[11] M. D. McIlroy, "A note on discrete representation of lines," *AT&T Tech. J.*, vol. 64, no. 2, pp. 481–490, 1984.

[12] M. Lindenbaum and J. Koplowitz, "A new parametrization of digital straight lines," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, pp. 847–852, 1991.

[13] A. Bruckstein, "The selfsimilarity of digital straight lines," in *Vision Geometry* (Melter, Rosenfeld, and Bhattacharya, Eds.). Contemporary Mathematics, AMS, 1991, pp. 1–20, vol. 119.

[14] J. Koplowitz, M. Lindenbaum, and A. Bruckstein, "On the number of digital straight lines on an $N \times N$ grid," *IEEE Trans. Inform. Theory*, vol. IT-36, pp. 192–197, 1990.

[15] C. Ronse, "A bibliography on digital and computational convexity (1961–1988)," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, pp. 181–190, 1989.

[16] B. A. Venkov, *Elementary Number Theory*, (translated and edited by H. Anderson). Groningen: Wolters-Noordhoff, 1970.

[17] K. Stolarsky, "Beatty sequences, continued fractions, and certain shift operators," *Canadian Math. Bull.*, vol. 19, no. 4, pp. 473–482, 1976.

[18] A. S. Fraenkel, M. Mushkin, and U. Tassa, "Determination of $[n\theta]$ from its sequence of differences," *Canadian Math. Bull.*, vol. 21, no. 4, pp. 441–446, 1978.

[19] R. L. Graham, S. Lin, and C. S. Lin, "Spectra of numbers," *Math. Mag.*, vol. 51, pp. 174–176, 1978.

[20] M. Boshernitzan and A.S. Fraenkel, "A linear algorithm for nonhomogeneous spectra of numbers," *J. Algorithms*, vol. 5, pp. 187–198, 1984.

[21] A. W. M. Smeulders and L. Dorst, "Decomposition of discrete curves into piecewise straight segments in linear time," in *Vision Geometry* (Melter, Rosenfeld and Bhattacharya, Eds.). Contemporary Mathematics, AMS, 1991, pp. 169–195, vol. 119.

[22] G. H. Hardy and E. M. Wright , *An Introduction to the Theory of Numbers*. Oxford: Oxford University Press, 1979.

[23] F. P. Preparata and M. I. Shamos, *Computational Geometry*, . Berlin: Springer Verlag, 1985.

[24] D. E. Knuth, *The Art of Computer Programming, Vol. 2, Seminumerical Algorithms*. Reading, MA: Addison-Wesley, 1969.

[25] L. Dorst and A. W. M. Smeulders, "Length estimators for digital contours," *Comput. Vision Graphics Image Processing*, vol. 40, pp. 311–333, 1987.

[26] J. O'Rourke, "An on-line algorithm for fitting straight lines between data ranges," *Commun. ACM*, vol. 24, no. 9, pp. 574–578, 1981.

Segmenting Handwritten Signatures at Their Perceptually Important Points

Jean-Jules Brault and Réjean Plamondon

Abstract— **This correspondence describes a new algorithm for segmenting continuous handwritten signatures sampled by a digitizer. The segmentation points are found by means of a two-step procedure. The principal step is to construct a function that weights the perceptual importance of every signature point according to its specific neighboring points. The second step points out the various local maxima of this function that correspond where the signature should be segmented. The method is well illustrated and tested on a number of signatures that require different kinds of segmentation decisions.**

Index Terms—**Corner detection, handwritten curve partitioning, handwritten signature, perceptual importance of an angle, segmentation.**

I. Introduction

The goal of an automatic signature verification (ASV) system is to confirm or invalidate the presumed identity of a signer from information obtained during execution of its signature. The techniques

Manuscript received December 20, 1989; revised February 8, 1993. This work was supported by grant CRSNG-A0915 and FCAR-AS-2655 and by the Ecole Polytechnique d Montréal. Recommended for acceptance by Editor-in-Chief A. K. Jain.

The authors are with Laboratoire Scribens, Département de Génie Electrique et Génie Informatique, Ecole Polytechnique de Montréal, Montréal, Canada H3C 3A7.

IEEE Log Number 9209988.