SOURCE (OR PART OF THE FOLLOWING SOURCE):
Type article
Title Vector code probability and metrication error in the representation of straight lines of finite length
Author(s) A.M. Vossepoel, A.W.M. Smeulders
Faculty UvA: Universiteitsbibliotheek
Year 1982

FULL BIBLIOGRAPHIC DETAILS:
http://hdl.handle.net/11245/1.424326

COMPUTER GRAPHICS AND IMAGE PROCESSING 20, 347–364 (1982)

Vector Code Probability and Metrication Error in the Representation of Straight Lines of Finite Length

A. M. VOSSEPOEL AND A. W. M. SMEULDERS*,†

*Department of Biomedical Information Processing and *Laboratory for Pathology, University Medical Center, Wassenaarseweg 62, 2333 AL Leiden, The Netherlands*

Received May 1, 1980; revised November 2, 1981

An unbiased estimate for the length of straight lines represented by an arbitrary number of discrete vector elements is derived from statistical evaluation of line segments randomly positioned on a grid. The computational method is independent of the connectivity of the grid, whether it is rectangular or hexagonal. Estimates for the variance of the length are also given. The length estimate may be used in combination with linearity conditions to evaluate the length of an arbitrary curved contour by polygonal approximation. The length of the original curve can then be estimated with greater accuracy than when existing methods are used. An alternative method for length estimation is also presented, based on least-squares approximation of infinitely long straight lines. For 8-connectivity, the alternative method gives a greater accuracy than similar existing methods. Figures are presented for both alternatives in comparison with existing methods.

1. INTRODUCTION

At first glance the computation of the length of a continuous line segment from its digital representation is rather straightforward. Until recently, the method most commonly used for computation of this length was that first described by Freeman [1]. In this method the computation of the length of a continuous line segment is replaced by the evaluation of the length of the digitized line segment. If we assume the digitized line segment to be represented by $n + 1$ grid points, then n vector elements will connect these grid points. The vector elements are usually coded according to Freeman [1]. All the n codes together are called the chain code string. The length of the digitized line segment in this method equals the sum of the lengths of the vector elements. In this paper a careful distinction should be made between the continuous line segment and the digitized line segment. The continuous line segment will sometimes be called line segment by way of abbreviation.

To find the length of a digitized line segment according to this method the length of each vector element is needed and will depend on the connectivity of the grid. If a rectangular grid is used and if only 4 neighbors are considered for each pixel, the grid has by definition a connectivity $c = 4$. If 8 neighbors are considered then the connectivity $c = 8$. Likewise, on a hexagonal grid $c = 6$, or $c = 12$. The length of a vector element equals the grid constant u for $c = 4$ or $c = 6$. The length of the even-coded vectors for $c = 8$ or $c = 12$ will also be u. For $c = 8$ the odd codes have a length $u\sqrt{2}$; for $c = 12$, $u\sqrt{3}$.

Let m denote the number of odd codes in a given chain code string with $c = 8$ consisting of n codes. In the naive method the length is then given by $n + m(\sqrt{2} - 1)$. It should be noted that to achieve the same density of grid points on a rectangular

†Present address: Department of Applied Physics, Delft University of Technology, The Netherlands.

0146-664X/82/120347-18$02.00/0

grid as on a hexagonal grid, the following relation should hold: $u_6 = u_4(\frac{4}{3})^{1/4}$. This relation may be derived by comparing the areas associated with one grid point in the two connectivities.

In 1978 Groen and Verbeek [2] demonstrated that the naive method for the computation of the segment length yields a biased estimate of the segment length. Via a rather complicated analysis of all possible intersections of a line segment with one grid cell they achieved unbiased estimates for straight line segments with $c = 8$ and $n = 1$. In the computation of the unbiased estimate the position of the relative entry height e in the grid cell and the orientation τ of the straight line segment are assumed to be random with respect to the grid. Given the probability density function of a straight line segment, determined by e and τ, integration over one grid cell now leads to the unbiased estimate. According to their method the length estimate is given by $1.059n + 0.124m$, also for $n > 1$. This segment length will be unbiased only under the assumption that there is no correlation between two subsequent vector elements. In practice this will rarely be the case. The method may be unrealistic in the assumption of uncorrelated chain codes, but basically it provides the concept of the statistical properties of single vector elements.

In 1979 Proffitt *et al.* presented a method of length estimation based on the properties of digitized lines of infinite length [3, 4]. It consists of computing the coefficients a and b while minimizing the error in the length defined by $an + bm$ for infinitely long lines. Because of the infinite length only the orientation, not the position of the line with respect to the grid, had to be considered as a parameter in their least-squares approach. For 8-connectivity they found $a = 0.948$ and $b = 0.392$.

The two methods of Groen [2] and Proffitt [3] lead, under different assumptions, to different results. Groen assumed no correlation between neighboring vectors ($n = 1$), and Proffitt assumed that $n = \infty$. If we have a digitized straight line segment of finite length ($n > 1$) neither method is unbiased. It is not clear which method should be preferred in a practical situation.

Apart from estimating the length of an infinite line segment Proffitt and Rosen have also introduced the corner count concept. A corner count n_c is defined as the number of consecutive and different code pairs. If the straight line segment is not nearly parallel to a grid axis such a corner will appear in the code string. As will be illustrated later, utilizing the corner count concept increases the accuracy of the length estimate. Proffitt arrives at a length estimate using n and n_c given by $0.948n - 0.278n_c$ for $c = 4$.

So far we have been concerned with length estimates of straight line segments. If we wish to estimate the length of an arbitrary curved line, we may approximate the code string by a polygon. The length of the curve may then be represented by the sum of the polygon side lengths. Methods are known in which the vertices of the polygonal approximation are chosen arbitrarily with a fixed number of codes between them [4], or according to an irregularity criterion as derived by Kulpa [5]. The length of the associated line segment is then computed as the Euclidean distance between the vertices. Quite clearly, for an arbitrary curved figure fixing the polygon side length a priori will be less appropriate than computing the places of the vertices from the code string. In the latter case, vertices may be found by applying linearity conditions to the code string. The linearity conditions are presented by Freeman [1] in an incomplete form and by Brons [6], both without a proof. A proof is given by Wu [7].

The linearity conditions can be summarized as follows:

–no more than two different code values are involved;
–the difference between these two code values is 1 or $c - 1$;
–the minority codes always appear isolated;
–no more than two different runlengths of majority codes are involved;
–the difference between these two runlengths is 1 or less;
–the minority runlengths of the majority codes always appear isolated;
–etc., starting iteration three conditions back, until only one runlength is involved, after replacing "majority codes" by "adjoining majority and minority runs".

Our aim in this paper is to derive an unbiased estimate for the length of a digitized line segment given n, m, and n_c. For each set of n, m, and n_c that may be generated by a straight line segment an unbiased estimate is computed. These estimates are found by generalization of the method of Groen for $n = 1$ and the method of Proffitt and Rosen for $n = \infty$ to finite values of n. In the estimates the corner count n_c is used to achieve an extra accuracy of the length estimate. In this paper the accuracy of the method presented will be compared with estimates of the form $na_n + mb_n + n_c c_n$, with coefficients a_n, b_n, and c_n which are found by solving the normal equations of a least-square fit. While solving these, combinations of a_n, b_n, and c_n will be found with greater accuracy than the ones given by Groen and Proffitt. Once the unbiased length estimates have been derived, our next aim is to approximate arbitrary curved figures by polygons. The length of the arbitrary figure will be found by summing the length estimates of the side lengths. Examples will be given for $c = 4$, $c = 6$, and $c = 8$; we omit $c = 12$ from the discussion, because this value is of little practical importance. Before we proceed to the computation of the estimates, we introduce the column concept in the next section. Use of the column concept makes the integration method independent of the connectivity.

2. DIGITIZATION

In the most commonly used coding scheme, the vector element in the direction of the positive x axis is coded 0. The other vector elements, at an angle $\tau = 2\pi j/c$ with the one coded 0 (τ taken counterclockwise only), are then coded j. In the following, the digitization of straight line segments is considered for directions τ only between those of vector elements coded 0 and 1, that is, $0 \leqslant \tau < 2\pi/c$.

Now we want to consider the complete set of straight line segments that result in n and only n codes upon digitization. To this end, we first define the concept of a column. An origin is supposed at a grid point (the open pixel center in Fig. 1a). Consider as the first line segment the one that connects the end points of the vectors that each consist of n vector elements coded 0 and 1, respectively. Consider as the second line segment the one that connects the end points of the vectors that each consist of $n + 1$ vector elements coded 0 and 1, respectively. Then column n is defined as the trapezoid between the parallel first and second line segments, and the vector elements numbered $n + 1$ coded 0 and 1, respectively. In Fig. 1a column 2 is indicated by shading.

The angle $\sigma(c)$ between the columns and the positive x axis only depends on the connectivity used: $\sigma(4) = \pi/4$, $\sigma(6) = \pi/3$, and $\sigma(8) = \pi/2$. This is also illustrated in Fig. 1a.

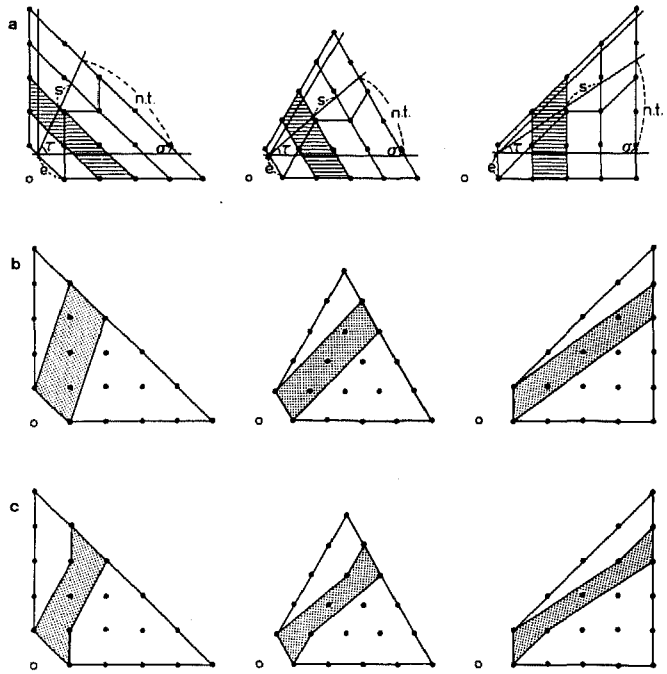

FIG. 1. Digitization of a straight line segment into four chain codes (1101, $n = 4$) in 4-, 6-, and 8-connectivity. (a) Illustration of the column concept (shading); (b) locus of the line segments that are digitized into three odd chain codes ($m = 3$); (c) same as (b), but with the additional constraint that both the first and the last chain code are odd ($k = 2$).

Among the different digitization methods that are used, we have adopted the one that connects the grid points nearest to the line segment when going along the intersected column boundaries in the direction of the positive x axis (see Fig. 1a). Alternatively, one could simply have taken the nearest grid points along the intersected column boundaries, or the ones in the opposite direction. But irrespective of the digitization method used, intersection of a straight line with n columns always results in n codes upon digitization, provided the orientation is within the limits, of course.

Now we define the entry window as the boundary between column 0 and column 1, and the exit window as the boundary between column n and column $n + 1$. An arbitrary straight line with orientation $\tau(0 \leqslant \tau < 2\pi/c)$, can always be made to intersect both the entry and exit windows by translating the grid origin over an integer number of vector elements. The proper orientation can be achieved by rotating the grid over an integer multiple of $\pi/4$ or $\pi/6$ in a rectangular or hexagonal grid, respectively. For double connectivities ($c = 8$ or $c = 12$) an additional reflection with respect to the direction of an odd-coded vector element may be necessary. Note that all these transformations of the grid do not change the position of any grid point, if considered anonymous.

The length of the straight line segment between the intersections with the column boundaries depends on the angle φ between the line segment and a line perpendicular to the parallel column boundaries. It also depends on the column width. The

ratio between this width and the grid constant u is called $q(c)$: $q(4) = \frac{1}{2}\sqrt{2}$, $q(6) = \frac{1}{2}\sqrt{3}$, and $q(8) = 1$. The ratio s between the intersected length and the grid constant u is then given by $s = q(c)/\cos\varphi$, and it is this length that generates exactly one code. So, in accordance with Groen and Verbeek [2], the number of codes per unit length is proportional to $\cos\varphi$, or inversely proportional to s. The latter proportionality turns out to be more convenient to use in further derivations. In s the connectivity is a parameter.

We define the entry height e' as the distance between the right-hand side of the entry window and the intersection between the entry window and the straight line. The relative entry height e is then defined as the ratio between e' and the entry window width (see Fig. 1a). In e also the connectivity is a parameter.

Likewise, we define the exit height $t' + e'$ as the distance between the right-hand side of the exit window and the intersection between the exit window and the straight line. The relative exit height t is then defined as the ratio between t' and the difference between the exit and entry window widths. This definition may seem rather artificial at first glance, but it provides the opportunity to transform functions of the variable τ into functions of the variable t, the former depending on the value of c, the latter not. Besides, the relations between t and τ are simple, for example, $t = \tan\tau$ for $c = 8$.

3. DOMAINS

By definition, the unbiased length estimate L is given by

$$L = \iint_{\text{domain}} s(\tau)\, p(e, \tau)\, d\tau\, de. \tag{1}$$

In this formula s denotes the ratio between the length of a line segment intersecting n columns, and $n \cdot u$, whereas p denotes the probability density function. Note that s does not depend on e, because the entry and exit windows are parallel. The integration will be described in the next section; here we focus on the integration domains.

The integration domains follow from all code strings representing a straight line that are considered equivalent. The most trivial approach is to consider all code strings with equal n as equivalent. Then the domain is defined by the full range of values of e and τ. But more information may be used in the length estimation than just the string length n. Let us consider the exit window depicted in Fig. 1a. We can easily think of confining the lines to a distinct part of the exit window, instead of treating all lines through the exit window as equivalent. A natural choice for these distinct parts would be the subwindow between two adjacent grid points on the exit window. In Fig. 1b the locus of all lines passing through the entry window and the exit subwindow is indicated. Upon digitizing, the chain code strings of all these lines have the number of odd codes m in common. The exit height within one subwindow is then given by $m \leqslant e + nt < m + 1$, with $0 \leqslant m < n$.

In Fig. 2a a diagram in $e + nt$ and t is given of the integration domain boundaries resulting when codes are considered equivalent if they have the same m. Within the ultimate boundaries imposed by $0 \leqslant e < 1$ and $0 \leqslant \tau < 2\pi/c$ (or $0 \leqslant t < 1 \Rightarrow e \leqslant e + nt < e + n$), the boundaries imposed by keeping m constant are given by

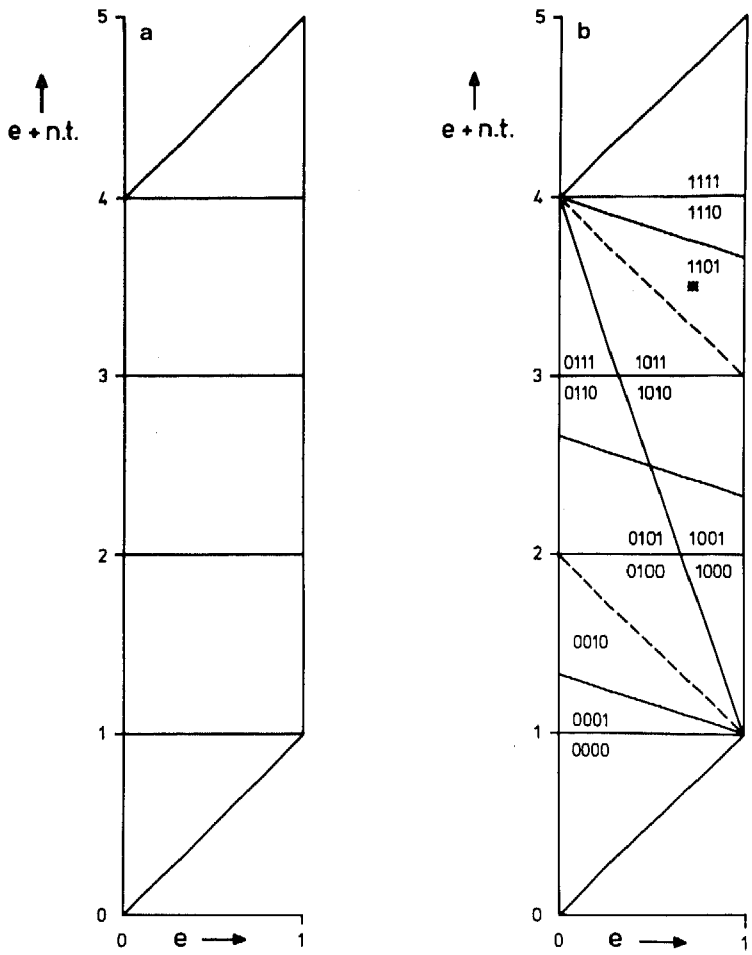

FIG. 2. Domains of chain code strings in the plane defined by e as abscissa and $e + nt$ as ordinate. The representation is independent of the connectivity. (a) Domains of chain code strings in which the number m of odd codes is constant. (b) Same as (a), but with the additional constraint that the number k of odd codes at both ends of the string is constant (cf. Fig. 1c). The chain code strings are indicated in the domains. Domain boundaries of strings with identical values of n, m, and k are indicated by dashed lines. The position of the continuous line segment in Fig. 1a is indicated by an asterisk.

the horizontal lines in Fig. 2a with

$$e + nt = m. \tag{2}$$

Proffitt *et al.* [3] stress the importance of using the corner count n_c when estimating the length of a line represented by a code string. When considering code strings of straight line segments the minority codes will appear isolated in the string in conformity with the linearity conditions. If both ends of the string consist of the majority codes, n_c will be twice the number of minority codes. But every minority code at the end of the string will reduce n_c by 1. So, for distinct values of n and m, n_c can have only three different values, which depend upon the parity of the codes at

the end of the chain code string. If we want to use the information provided by n_c for a further reduction of the integration domain (thus increasing the accuracy of the length estimates) we can just as well use k, the total number of odd codes at each end of the string instead of n_c. In the frame of reference provided by Fig. 1, in which the codes can only have values 0 or 1, k is defined as

$$k = \text{code}(1) + \text{code}(n).$$

Consequently, $k = 0$, 1, or 2. Among the set of all line segments the value of code(1) is determined by considering the exit height $(e + t)$ after intersection with the first column:

$$e + t < 1 \Rightarrow \text{code}(1) = 0, \qquad \text{and} \qquad e + t \geqslant 1 \Rightarrow \text{code}(1) = 1. \tag{3}$$

Likewise, the value of code(n) is determined by the difference between the digitized exit height after n columns and the one after $n - 1$ columns. If both exit subwindows are described by the same value of m, code(n) = 0; otherwise code(n) = 1. The value of m that describes a subwindow is given by $m = [e + nt]$, in which the brackets denote the entier (truncation to integer) function. The value of the entier function will change at $e + nt = m$, which is Eq. (2) again, and at $e + (n - 1)t = m$.

In Fig. 1c the reduction of the locus of all lines with distinct n and m by taking $k = 2$ is depicted. In Fig. 2b the boundaries resulting from the introduction of k are added to Fig. 2a:

$$\begin{array}{llll} e + t = 1 & \text{or} & e + nt = n - (n - 1)e & \text{(steep diagonal)} \\ e + (n - 1)t = m & \text{or} & e + nt = \dfrac{nm}{n - 1} - \dfrac{e}{n - 1} & \text{(sloping lines).} \end{array} \tag{4}$$

So in Fig. 2b every closed domain corresponds to one combination of n, m, and k.

The equations of the boundaries can be written in a much simpler form as functions $e(t)$. Equations (2) and (4) then become

$$\begin{array}{ll} e = m - nt & \text{for } 1 \leqslant m \leqslant n \\ e = m - (n - 1)t & \text{for } 1 \leqslant m < n \\ e = 1 - t. & \end{array} \tag{5}$$

In Fig. 3 the boundaries are depicted in the e–t plane, for $n = 4$ again.

In the evaluation of the integral of Eq. (1)—as will be explained in the next section—we need a description of the domains by the values of the relative exit height t and of the change r in direction coefficient, both at the corners of each integration domain. To find general expressions for t and r we observe that the integration domains generally appear in sets of four quadrangular domains with equal values of m. In Fig. 3 one such set may be seen at the center of the figure (i.e., the four domains that have ($e = 0.5$, $t = 0.5$) as one corner). The upper left domain of such a set is always for $k = 0$, the lower right domain for $k = 2$. The upper right and lower left domains are always both for $k = 1$. Note that in Fig. 3 there is only one such set of domains, with $m = 2$, because $n = 4$. For larger values of n there are more—$(n - 3)$, in general—such sets. For $m = n - 1$, the set is degenerate, because

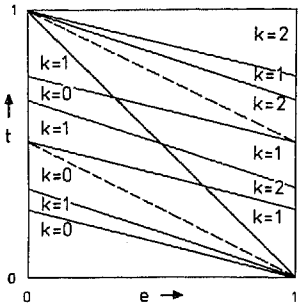

FIG. 3. Domains of chain code strings with constant values of n, m, and k in the e–t plane ($n = 4$, cf. Fig. 2b). The representation is independent of the connectivity.

the domain for $k = 0$ has vanished. Likewise, the domain for $k = 2$ has vanished from the set for $m = 1$. In these two cases two corners of each domain for $k = 1$ coincide. For the extreme values of m ($m = 0$ and $m = n$) the sets comprise just one domain each, requiring special treatment, as will be explained in the next section. So, for $0 < m < n$, each domain can be viewed as one of a set of four (or three), according to the value of m. The values of t and r at the corners of the domains are presented in Table 1.

In the next section the planar integral is taken over every such closed domain resulting in a different length estimate for every combination of n, m, and k. The area of a domain together with the probability density function of the line segment given in e and t determine the variance in the length estimate of chain code strings with corresponding n, m, and k.

TABLE 1

Values of the Relative Exit Height t and Change in Direction Coefficient r, Both at the Four Corners of Each of the Four Integration Domains of a Set with a Distinct Value of m, $0 < m < n$

Domain quadrant	k	Corner quadrant	t	r	Domain quadrant	k	Corner quadrant	t	r
2	0	1	$\frac{m}{n-1}$	$1-n$	1	1	1	$\frac{m}{n}$	$-n$
2	0	2	$\frac{m+1}{n}$	n	1	1	2	$\frac{m}{n-1}$	$n-1$
2	0	3	$\frac{m}{n-1}$	$1-n$	1	1	3	$\frac{m-1}{n-2}$	$2-n$
2	0	4	$\frac{m-1}{n-2}$	$n-2$	1	1	4	$\frac{m-1}{n-1}$	$n-1$
3	1	1	$\frac{m-1}{n-2}$	$2-n$	4	2	1	$\frac{m-1}{n-1}$	$1-n$
3	1	2	$\frac{m}{n-1}$	$n-1$	4	2	2	$\frac{m-1}{n-2}$	$n-2$
3	1	3	$\frac{m}{n}$	$-n$	4	2	3	$\frac{m-1}{n-1}$	$1-n$
3	1	4	$\frac{m-1}{n-1}$	$n-1$	4	2	4	$\frac{m-1}{n}$	n

4. INTEGRATION

The unbiased length estimate L is given in Eq. (1). Instead of evaluating that equation, we compute L and its variance from $L(n, m, k) = G_1/G_0$ and

$$\operatorname{var} L(n, m, k) = G_2/G_0 - (G_1/G_0)^2 \tag{6}$$

in which the moment functions G_i are given by

$$G_i(n, m, k) = \iint_{\text{domain}} s(\tau)^i h_0(\tau, e)\, d\tau\, de. \tag{7}$$

In this formula $h_0(\tau, e)$ is the unnormalized weight function. The integral over τ and e should be performed for each domain, of which the boundary conditions are computed in the previous section. For every domain, with given n, m, and k, a different value of L will result. The variable τ is rather unpractical in the integration because $s(\tau)$ depends of the connectivity in a complicated way. It also presents complicated relations between e and τ at the integration domain boundaries, as opposed to the ones given in Eq. (5). To avoid these complications we use in the integration the variable t instead:

$$G_i(n, m, k) = \iint_{\text{domain}} s(t)^i h_0(t, e)(dt/d\tau)^{-1}\, dt\, de. \tag{8}$$

From the sine rule follows

$$t = \frac{\sin\tau}{\sin(\tau + \sigma)} \Rightarrow \frac{dt}{d\tau} = \frac{\sin\sigma}{\sin^2(\tau + \sigma)}. \tag{9}$$

Applying the sine rule once more gives

$$\sin(\tau + \sigma) = \sin\sigma/s$$

and substituting this into Eq. (9) results in

$$\frac{dt}{d\tau} = \frac{s^2}{\sin\sigma}. \tag{10}$$

Now, application of the cosine rule gives

$$s(t) = (t^2 + 1 - 2t\cos\sigma)^{1/2} \tag{11}$$

The weight function h_0 is adopted from the considerations of Groen and Verbeek [2], as already mentioned in Section 2. Assuming n and c constant, the code generating probability per unit length of a straight line segment is inversely proportional to its length s:

$$h_0 = 1/s, \qquad \text{independent of } e. \tag{12}$$

Substituting (12), (11), and (10) in (8) yields

$$G_i(n, m, k) = \iint_{\text{domain}} s(t)^{i-3} \sin\sigma \, dt \, de. \tag{13}$$

The integrand is a function of t only, and the domain boundaries are linear relations in e and t (see Eq. (5)). The evaluation of an integral of this type is presented in generalized form in Appendix A. The result is

$$G_i(n, m, k) = \sum_{j=1}^{4} r_j F_i(t_j) \tag{14}$$

in which the values t_j denote the ordinates of the four corner points of the integration domain. The values r_j denote the change in the direction coefficient of the boundary of the integration domain at each corner, going in a counterclockwise direction along the boundary. The values of t_j and r_j depend on n, m, and k and follow from the scheme presented in the previous section.

Finally, the functions $F_i(t)$ in Eq. (14) are found by

$$d^2F_i(t)/dt^2 = s(t)^{i-3}\sin\sigma \qquad \{= h_0(t)s(t)^{i-2}\sin\sigma\}.$$

Integrating this expression twice over t gives

$$\begin{aligned}
F_0(t) &= s(t)/\sin\sigma \\
F_1(t) &= n((t - \cos\sigma)\arctan((t - \cos\sigma)/\sin\sigma) - \sin\sigma\log(s(t))) \\
F_2(t) &= n^2\sin\sigma((t - \cos\sigma)\log(s(t) + t - \cos\sigma) - s(t)).
\end{aligned}$$

For $0 < m < n$ the following expressions for $G_i(n, m, k)$ result

$$\begin{aligned}
G_i(n, m, 0) &= 2(1 - n)F_i\left(\frac{m}{n-1}\right) + nF_i\left(\frac{m+1}{n}\right) + (n-2)F_i\left(\frac{m-1}{n-2}\right) \\
G_i(n, m, 1) &= 2\left\{-nF_i\left(\frac{m}{n}\right) + (2-n)F_i\left(\frac{m-1}{n-2}\right)\right. \\
&\qquad \left. + (n-1)\left(F_i\left(\frac{m}{n-1}\right) + F_i\left(\frac{m-1}{n-1}\right)\right)\right\} \\
G_i(n, m, 2) &= 2(1 - n)F_i\left(\frac{m-1}{n-1}\right) + nF_i\left(\frac{m-1}{n}\right) + (n-2)F_i\left(\frac{m-1}{n-2}\right).
\end{aligned}$$

For $m = 0$ or $m = n$, the integration has to be performed according to Appendix B, resulting in

$$\begin{aligned}
G_i(n, 0, 0) &= nF_i(1/n) - nF_i(0) - H_i(0) \\
G_i(n, n, 2) &= H_i(1) - nF_i(1) + nF_i((n-1)/n)
\end{aligned}$$

in which the functions $H_i(t)$ are the result of integrating $h_0(t)s(t)^{i-2}\sin\sigma$ once

over t

$$H_0(t) = (t - \cos\sigma)/(s(t)\sin\sigma)$$
$$H_1(t) = n\arctan\{(t - \cos\sigma)/\sin\sigma\}$$
$$H_2(t) = n^2 \sin\sigma \log(s(t) + t - \cos\sigma).$$

Evaluation of these expressions for distinct values of n, m, and k, and substitution of these values in Eq. (6) then gives the unbiased length estimate $L(n, m, k)$ and its variance. The resulting values are summarized in Table 2 for $1 \leqslant n \leqslant 4$. From this table it appears that the length estimation of coded linear segments is most accurate with $c = 6$ for short segments.

A graphic representation of some results is presented in Fig. 4. Essentially, this figure renders the domain areas as given by Eq. (5), similarly to Fig. 3. The ordinate

TABLE 2

Relative Probability or Weight w, Length s ($= L/n$), and Variation Coefficient d for Some Numbers of Codes n, as a Function of n, the Number of Odd Codes m, and the Number k of Odd Codes at Both Ends of the Code String

				Connectivity = 4			Connectivity = 6			Connectivity = 8		
n	m	k	n_c	w	s	d (%)	w	s	d (%)	w	s	d (%)
1	0	0	0	0.500	0.785	10.2	0.500	0.907	4.3	0.586	1.059[a]	7.0
1	1	2	0	0.500	0.785	10.2	0.500	0.907	4.3	0.414	1.183[a]	10.0
1		total		1.000	0.785	10.2	1.000	0.907	4.3	1.000	1.111	8.4
2	0	0	0	0.207	0.837	10.0	0.232	0.930	4.2	0.334	1.019	2.2
2	1	1	1	0.586	0.749	7.0	0.536	0.887	2.9	0.504	1.113	7.5
2	2	2	0	0.207	0.837	10.0	0.232	0.930	4.2	0.162	1.292	5.9
2		total		1.000	0.785	8.4	1.000	0.907	3.5	1.000	1.111	5.9
3	0	0	0	0.118	0.885	7.6	0.146	0.949	3.3	0.229	1.009	1.0
3	1	1	1	0.178	0.772	7.2	0.173	0.897	3.0	0.209	1.041	2.5
3	1	0	2	0.204	0.739	6.4	0.182	0.882	2.6	0.189	1.075	4.4
3	2	2	2	0.204	0.739	6.4	0.182	0.882	2.6	0.148	1.165	6.1
3	2	1	1	0.178	0.772	7.2	0.173	0.897	3.0	0.127	1.228	4.8
3	3	2	0	0.118	0.885	7.6	0.146	0.949	3.3	0.099	1.333	4.0
3		total		1.000	0.785	7.0	1.000	0.907	2.9	1.000	1.111	3.9
4	0	0	0	0.081	0.914	5.9	0.106	0.961	2.6	0.174	1.005	0.6
4	1	1	1	0.074	0.823	6.1	0.080	0.919	2.6	0.111	1.021	1.2
4	1	0	2	0.178	0.772	7.2	0.173	0.897	3.0	0.209	1.041	2.5
4	2	2	2	0.052	0.736	2.5	0.046	0.878	0.9	0.049	1.063	1.7
4	2	1	3	0.229	0.713	1.0	0.191	0.869	0.4	0.169	1.117	2.7
4	2	0	2	0.052	0.736	2.5	0.046	0.878	0.9	0.035	1.186	2.4
4	3	2	2	0.178	0.772	7.2	0.173	0.897	3.0	0.127	1.228	4.8
4	3	1	1	0.074	0.823	6.1	0.080	0.919	2.6	0.056	1.281	3.4
4	4	2	0	0.081	0.914	5.9	0.106	0.961	2.6	0.071	1.354	3.0
4		total		1.000	0.785	5.5	1.000	0.907	2.4	1.000	1.111	2.7

[a] Cf. [2].

Note. The number of corners is denoted by n_c.

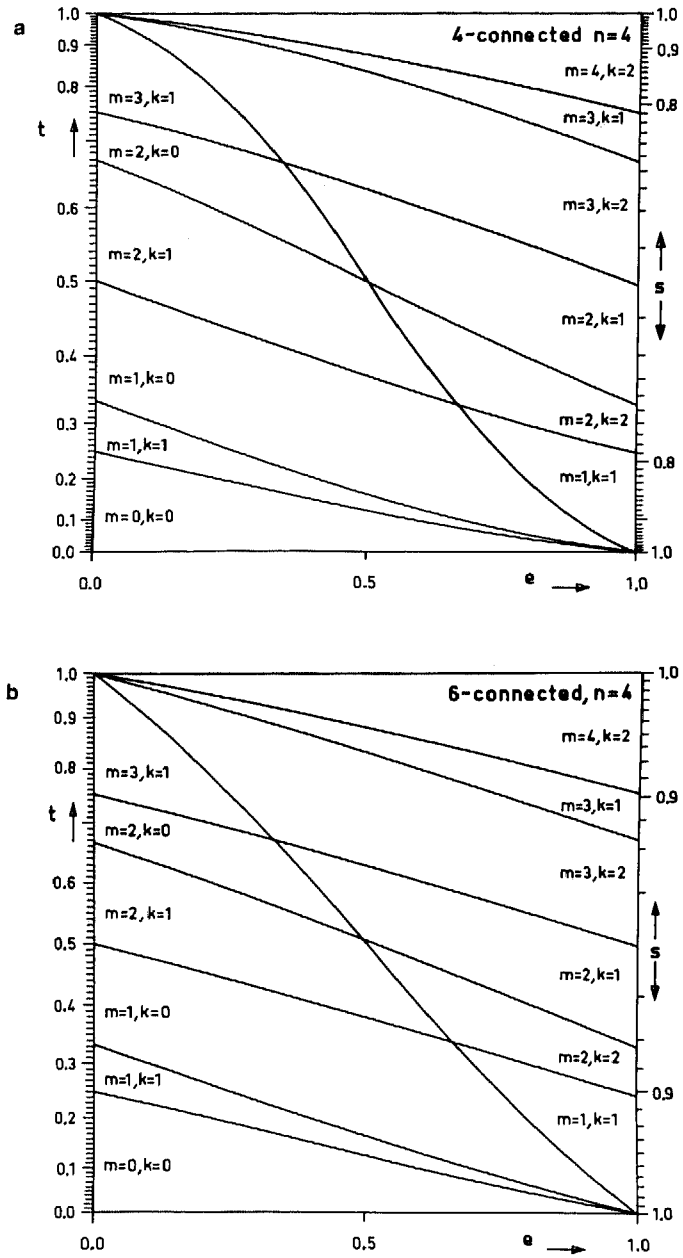

FIG. 4. Same as Fig. 3, but the vertical axis is transformed such that the domain areas are proportional to the probability of the chain code strings. Vertical (nonlinear) scales indicate normalized segment length s and relative exit height t (cf. text and Fig. 1a). The connectivity is indicated in each subfigure.

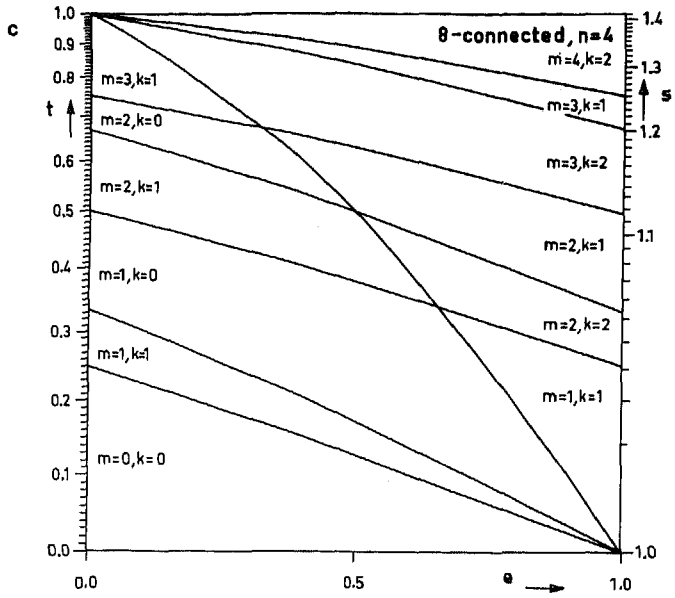

FIG. 4.–*Continued.*

t is transformed into

$$\frac{H_0(t) - H_0(0)}{H_0(1) - H_0(0)}$$

with $H_0(t) = dF_0(t)/dt$ again.

By transforming the t axis in this way the area of each domain becomes proportional to the integrated weight function $G_0(n, m, k)$, that is, proportional to the fraction of random oriented line segments with distinct values of m and k, given n.

The values of t are still indicated on the vertical axis ($e = 0$) on a nonlinear scale. On the boundary given by $e = 1$, some values of $s(t)$ are indicated, reflecting the decreasing probability of segments of increasing length.

5. COMPARISON WITH OTHER METHODS

Both Groen and Verbeek [2] and Proffitt and Rosen [3, 4] compute the segment length K by a relation linear in n, m, and n_c. In this section we will repeat the evaluation of the coefficients of the linear relation by a least-square approximation over m and n_c, for a distinct value of n. K is given by

$$K(n, m, n_c) = na_n + mb_n + n_c c_n$$

in which a_n represents the length assigned to all codes, b_n denotes the length correction for all odd codes, and c_n the correction for the corner count n_c.

Given the functions $G_0(n, m, k)$ and $L(n, m, k)$ derived above, the normal equations used in the least-squares approximation are

$$\sum (nG_0(K/L - 1)) = 0$$
$$\sum (mG_0(K/L - 1)) = 0$$
$$\sum (n_c G_0(K/L - 1)) = 0,$$

in which the summations are performed over the different values of m and n_c, given n. The arguments of the functions G_0, K, and L have been omitted for the sake of clarity. Solution of these equations leads to the values for a_n, b_n, and c_n.

For 4- or 6-connectivity we remove the second equation from the set and put $b_n = 0$. For $n = \infty$ and $c = 4$ the same result is found as in [3]. In 8-connectivity we may account for the corner count, but fix the correction of the odd codes a priori by $b_n = a_n(\sqrt{2} - 1)$. The second equation may then be removed from the normal equations. Instead of removing the second equation in 8-connectivity, the third equation may be removed, that is, the corner count will not be taken into account in the length computation ($c_n = 0$). By this procedure for $n = 1$ the same result is found as Groen's [2], whereas for $n = \infty$ the same result is found as Proffitt's [3]. In 8-connectivity removing an equation from the set will lead to a larger average error of estimate. Therefore we also evaluate all three normal equations for $c = 8$ without imposing any constraint on b_n or c_n.

In Table 3 the values of a_n, b_n, and c_n are presented for $n = 1000$. This number of codes is large enough to make the results comparable with those described (for $c = 4$) for infinite n. The results for 6- and 8-connectivity are also presented using the constraints mentioned before. One may see from the table that the constraint $c_n = 0$ in 8-connectivity results in the same values for the coefficients per code a_n and in the same variation coefficient d_n as in 4-connectivity. Since the line segment rendering 1000 codes is on average much longer in 8- than in 4-connectivity, accuracy is lost in $c = 8$ by removing the corner count from the length computation.

When we consider the variation coefficients d_n' in Table 3, the accuracy of long linear segments seems optimal with $c = 8$. However, for large n the average absolute

TABLE 3

Coefficients a_n (Average Length per Code), b_n (Odd Code Correction), and c_n (Corner Correction) Used in the Computation of the Segment Length $K = a_n n + b_n m + c_n n_c$ for $n = 1000$

Connectivity	Constraints	a_n	b_n	c_n	d_n(%)	d_n'(%)	e_n'
4	$b_n = 0$	0.948[a]	0.000	−0.278[a]	2.3[a]	0.021	0.184
6	$b_n = 0$	0.977	0.000	−0.131	1.0	0.010	0.096
8	$b_n = a_n(\sqrt{2} - 1)$	0.984	0.408	−0.085	0.8	0.009	0.114
8	$c_n = 0$	0.948[a]	0.392[a]	0.000	2.3[a]	0.009	0.114
8	–	0.980	0.426	−0.091	0.7	0.009	0.114

[a] Cf. [3].

Note. The term d_n denotes the resulting average variation coefficient of K, d_n' the same for the unbiased length estimate L, and e_n' the average absolute error in L.

error e'_n in the segment length is decreasing toward a constant with increasing number of codes. This constant value is smaller for $c = 6$ than for $c = 8$, even after application of the necessary correction factor $(4/3)^{1/4}$ for $c = 6$ to obtain equal grid densities.

6. POLYGON LENGTH

By applying the linearity conditions mentioned in Section 1 to an arbitrary code string one may construct a polygonal representation of this string. Starting with the first code, one code is added to a string and the linearity conditions are applied, until the string does not fulfill the conditions any more. The code that made the previous string fail to meet the linearity conditions may in turn be taken as the first code of the next string. The process may be repeated until the end of the string is reached. For closed contours that contain a "sharp" corner, that is, two adjacent codes representing nonadjacent directions, it makes sense to start at this sharp corner.

The vertex-finding process is illustrated in Fig. 5 for a closed 4-connected contour and its 8-connected counterpart. The figure has been generated by boundary quantization of a hand-drawn circular object. If no sharp corner is present, the first corner where a third different code appears after an otherwise linear code string is taken as a starting point (P or Q). The result depends strongly on the (arbitrary) choice of the entry point E, so the starting point is searched for only beyond the first corner after the entry point. In the given example the 4-connected contour, consisting of 66 codes (vector elements), has a length of 52.58 (± 0.62) starting at P and going clockwise, or 53.11 (± 0.61) starting at Q and going counterclockwise. Both times, 8 linear segments were found (see Fig. 5). In the 8-connected contour, consisting of 44 codes, 9 linear segments were found, resulting in lengths 52.46 (± 0.50) and 52.62 (± 0.44) starting at P and Q, respectively. The errors were computed by adding the theoretical variances of the individual segments. The contour lengths have been corrected for object boundary quantization by addition of

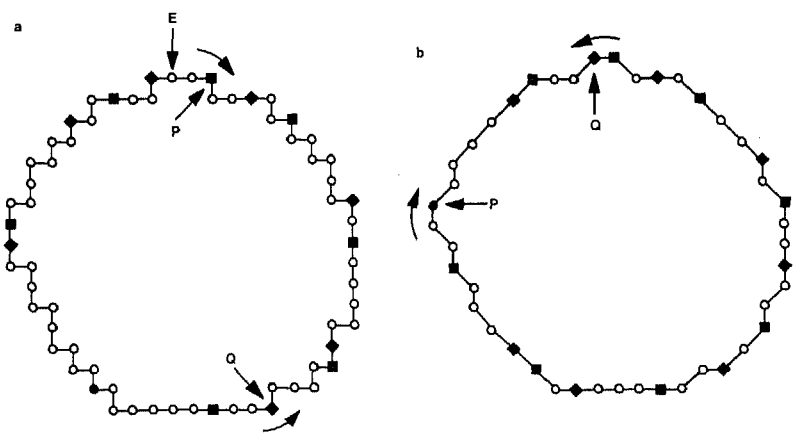

FIG. 5. Polygonal representation of a digitized circular object in (a) 4- and (b) 8-connectivity. The entry point of the contour is E (in Fig. 5b E coincides with Q). The P and Q are starting points for clockwise (■) and counterclockwise (◆) searches for vertices, respectively. A ● indicates a vertex found in both directions.

APPENDIX A

Let us consider an integration domain A in the $e - t$ plane, limited by the boundary $b(e, t)$. Let us assume that the boundary consists of four straight line segments, that is, on each segment between the corner points i and j ($j = i + 1$) the direction coefficient

$$v_{ij} = \left\{ \frac{de(t)}{dt} \right\}_{b(e,t)}$$

is a constant, except if a boundary is described by a constant value for e. In that case, we have to resort to the integration method given in Appendix B.

We want to compute

$$G = \int\int_A h(t)\, dt\, de.$$

According to Green's theorem we can write this as

$$G = \oint_{b(e,t)} H(t)\, de,$$

in which $H(t)$ is defined by $h(t) = dH(t)/dt$. Note that the boundary is supposed to be followed counterclockwise. Now we may write

$$G = \oint_{b(e,t)} H(t) \left(\frac{de(t)}{dt} \right)_{b(e,t)} dt$$

as long as the direction coefficients v_{ij} exist. But in the assumed case, the coefficients v_{ij} are constants. So, if we define $F(t)$ by $H(t) = dF/dt$, we may write

$$G = \sum_{i=1,(j=i+1)}^{4} v_{ij}\big(F(t_j) - F(t_i)\big),$$

or

$$G = \sum_{j=1,(i=j-1),(k=j+1)}^{4} r_j F(t_j), \qquad \text{with} \quad r_j = v_{ij} - v_{jk}.$$

APPENDIX B

In practice, there are two cases in which the direction coefficient $(de/dt)_b$ is not defined, for $t = 0$ and $t = 1$, respectively. In these cases the planar integral cannot be solved by the method given in Appendix A.

$$\int_0^1 \int_0^{t(e)} h(t)\, dt\, de = \int_0^1 (H(t(e)) - H(0))\, de$$
$$= -H(0) + \int_{t(0)}^{t(1)} H(t)(de/dt)_b\, dt = -H(0) + (de/dt)_b\{F(t(1)) - F(t(0))\}.$$

Likewise

$$\int_0^1 \int_{t(e)}^1 h(t)\, dt\, de = \int_0^1 (H(1) - H(t(e)))\, de$$

$$= H(1) - \int_{t(0)}^{t(1)} H(t)(de/dt)_b\, dt$$

$$= H(1) - (de/dt)_b \{F(t(1)) - F(t(0))\}.$$

ACKNOWLEDGMENTS

We are indebted to Dr. F. C. A. Groen for creating the opportunity to write this paper. We thank him, Dr. R. P. W. Duin, and Ir. L. Dorst for their constructive criticism in reviewing the manuscript.

REFERENCES

1. H. Freeman, Boundary encoding and processing, In *Picture Processing and Psychopictorics* (B. S. Lipkin and A. Rosenfeld, Eds.), pp. 241–266, Academic Press, New York, 1970.
2. F. C. A. Groen and P. W. Verbeek, Freeman-code probabilities of object boundary quantized contours, *Computer Graphics and Image Processing* **7**, 1978, 391–402.
3. D. Proffitt and D. Rosen, Metrication errors and coding efficiency of chain-encoding schemes for the representation of lines and edges, *Computer Graphics and Image Processing* **10**, 1979, 318–332.
4. T. J. Ellis, D. Proffitt, D. Rosen, and W. Rutkowski, Measurement of the lengths of digitized curved lines. *Computer Graphics and Image Processing* **10**, 1979, 333–347.
5. Z. Kulpa, Area and perimeter measurement of blobs in discrete binary pictures, *Computer Graphics and Image Processing* **6**, 1977, 434–451.
6. R. Brons, Linguistic methods for the description of a straight line on a grid, *Computer Graphics and Image Processing* **3**, 1974, 48–62.
7. L. D. Wu, On the Freeman's conjecture about the chain code of a line. *Proceedings of the Fifth International Conference on Pattern Recognition, Miami Beach, Fla., December 1–4, 1980*, pp. 32–34, IEEE, New York, 1980.
8. A. W. M. Smeulders, A. M. Vossepoel, J. Vrolijk, J. S. Ploem, and C. J. Cornelisse, Some shape parameters for cell recognition, In *Pattern Recognition in Practice* (E. S. Gelsema and L. N. Kanal, Eds.), pp. 131–142, North-Holland, Amsterdam, 1980.
9. A. M. Vossepoel and A. W. M. Smeulders, Statistical properties of digitised line segments, In *Computer Graphics 81* (Conference Proceedings, London, October 27–29, 1981), pp. 471–483, Online, Northwood Hills, UK, 1981.