Length estimators compared

Dorst, L.; Smeulders, A.W.M.

Published in:
Pattern recognition in practice II: proceedings of an International Workshop held in Amsterdam, June 19-21, 1985

PATTERN RECOGNITION IN PRACTICE II
E.S. Gelsema and L.N. Kanal (Editors)
Elsevier Science Publishers B.V. (North-Holland), 1986

LENGTH ESTIMATORS COMPARED

Leo Dorst
Pattern Recognition Group
Department of Applied Physics
Delft University of Technology
Lorentzweg 1
2628 CJ DELFT
The Netherlands

Arnold W.M. Smeulders
Pathologisch Instituut
De Boelelaan 1117
1007 MB AMSTERDAM
The Netherlands

Six methods to measure the length corresponding to a chaincode string are compared with respect to accuracy and algorithmic complexity. The comparison is performed for straight chaincode strings, for which the optimal method of length measurement is known. A sub-optimal method, the so-called 'corner-count method' seems to be the best to use in practice. It has an accuracy ranging from 3% for a sampling density of 5 points per unit length to .6% for infinite sampling density.

INTRODUCTION

Measuring geometrical properties of objects in digitized images has been important since the start of image processing. Area, contour length, and orientation are often parameters of interest, be it in industrial inspection, robot vision, biomedical research, or remote sensing. But not for all geometrical parameters is there an obviously best way in which they should be measured from the digital image. For length measurement, for instance, many methods have been proposed [1]-[7]. In this paper, we will compare these methods, both in terms of accuracy and in terms of algorithmic complexity. The comparison will be performed for the measurement of the length of a straight contour. For this situation the optimal solution is known [3], and thus the different methods can be gauged in an absolute sense.

LENGTH MEASUREMENT METHODS

Since more than 15 years people have tackled the problem of measuring the length of a digitized straight boundary. The methods that will be compared in this paper are now presented in chronological order.

In the dawn of image processing for measurement purposes, Freeman [4] proposed the chain code scheme to encode contours in a convenient way (fig. 1). This scheme is still in use today. Note that there are even codes (horizontal and vertical) and odd codes (diagonal). A length can be attributed to a chain coded contour by the formula

$$\ell_F = 1.n_e + \sqrt{2}\, n_o = 1.0\, n_e + 1.414\, n_o \qquad (1)$$

where n_e is the number of even codes in the chaincode of the contour, and n_o the number of odd codes. If this length estimate is used to measure the distance to a fixed grid point it can be compared to the Euclidean distance. In fig 2a the points of equal ℓ_F-distance to a point (the 'ℓ_F-circle') are drawn together with a Euclidean circle. It is seen that the ℓ_F distance is always smaller than or equal to the Euclidean distance. The ℓ_F-distance thus does not yield an unbiased estimate of the length of a straight line: it is consistently too small.

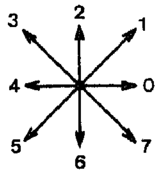

figure 1. Freeman's chaincode scheme

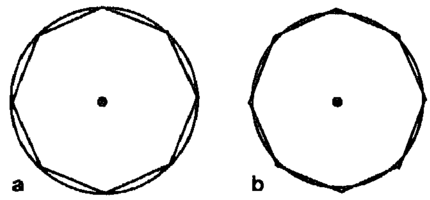

figure 2. A Euclidean circle compared to an ℓ_F-circle (a) and a ℓ_P-circle.

Formula (1) has the advantage of being easy to compute, and a number of authors concentrated on estimators of the form:

$$\ell = an_e + bn_o \ . \tag{2}$$

Groen and Verbeek [5] remarked that the ℓ_F formula does not take into account that the a-priori probabilities of odd and even codes in the string of a straight line are unequal. For an ensemble of lines that is uniformly distributed in the polar parameters r and ϕ [3], they found that the coefficients a = 1.059 and b = 1.183 in (2) yield a length estimate which is unbiased per chaincode. This leads to an estimator

$$\ell_G = 1.059\ n_e + 1.183\ n_o \tag{3}$$

These coefficients are the expected length of an even and odd chaincode, respectively.
Proffitt and Rosen [6] had also noticed that the ℓ_F-estimator is biased, and performed a calculation similar to Groen and Verbeek. They also calculated the expected length of an even and an odd chaincode in infinitely long strings and obtained a = .948 and b = 1.340. The resulting estimator thus is

$$\ell_P = .948\ n_o + 1.340\ n_e. \tag{4}$$

(In [6] the coefficients were originally calculated for 4-connective chain codes; in [7] they were calculated by the same method for 8-connectivity. This results in (4)). The difference with (3) is that Proffitt and Rosen considered the a-priori probabilities of the even and odd codes in infinitely long lines, whereas Groen and Verbeek calculated the a-priori probabilities considering 1 column of the grid.
Note that (4) can be rewritten as

$$\ell_P = .948\ (1.0 n_e + 1.414\ n_o) = .948\ \ell_F \tag{5}$$

which is a rescaling of the Freeman estimator, to render it unbiased in the limit of infinitely long lines (fig. 2b).

In [6] the authors also tried to improve the accuracy by introducing an extra parameter in the estimator. In this extended version

$$\ell = a\ n_e + b\ n_o + c\ n_c, \tag{6}$$

the parameter n_c is a parameter called 'corner count', which is the number of transitions odd/even and even/odd in the chaincode string. They performed the calculation for 4-connectivity. A similar calculation was later done by Vossepoel & Smeulders in [7] for 8-connectivity and results in:

$$\ell_C = .980\ n_e + 1.406\ n_o - .091\ n_c \qquad (7)$$

The use of n_o and n_e as parameters means that only begin- and endpoint of a line contribute to the length estimator. In [7] it was shown that the use of n_c effectively means that the points in the second and second last column also contribute, resulting in a considerable gain in accuracy.

Also in [7], Vossepoel and Smeulders noticed the difference between (3) and (4) and concluded that the coefficients in the estimators for the length of a discrete line should depend on the number of chaincodes in the string. They conjectured that optimal estimators could only result if each line was endowed with its own specific estimate. This requires a set of parameters characterizing an arbitrary chaincode string uniquely. They chose n_e, n_o and n_c, and thus arrived at estimators of the form

$$\ell_V = f(n_e,\ n_o, n_c) \qquad (8)$$

The functions f are complicated and can be found in [7].

However, there is no one-to-one relationship between the triple of parameters (n_e, n_o, n_c) and a straight chaincode string : each string leads to a unique set of parameters, but the converse is not true. This implies that Vossepoel and Smeulders did not quite reach their own goal, and that their estimators could still be improved. This was done by Dorst and Smeulders [2]. In their paper they derived a set of 4 integer parameters (n,q,p,s) that characterizes an arbitrary straight string uniquely. In a future paper [3] these parameters are used to make length estimators that are optimal for the straight string under consideration. These estimators, which are of the form

$$\ell_D = g(n,q,p,s) \qquad (9)$$

(where g is a known but complicated function specified in [3]) can be shown to be BLUE, i.e. Best (in the sense of minimal mean square error), Linear and Unbiased. For a straight string, no better estimate of the length can be found.

Recently, Borgefors [1] published a method to perform the distance transform on a binary image. She uses integer calculations and computes the so-called Chamfer distances:

$$\ell_B = an_e + bn_o \qquad \text{, a and b integer} \qquad (10)$$

which afterwards is rescaled by a. The ratio b/a is an approximation of $\sqrt{2}$. We have not yet evaluated this estimator, but it is expected to behave like ℓ_F if the scale factor is taken to be a (as is proposed in [7]) and like ℓ_P if the scale factor is improved to remove the bias.

THE COMPARISON FOR STRAIGHT BOUNDARIES

Most of the methods presented in the previous section were derived originally for digitized straight lines, but are generally used for arbitrary chaincode strings. The comparison in this paper will be performed on digitized straight lines, for two reasons. The first is that it seems proper to use the estimators for the goal they were meant for. The second is that the optimal solution for straight lines is known (the BLUE estimators of [3]), so one can compare the estimators to the theoretically best possible. The major drawback of this choice is of course that the majority of objects of interest have no straight boundaries, and hence care should be taken in applying these results to arbitrary strings.

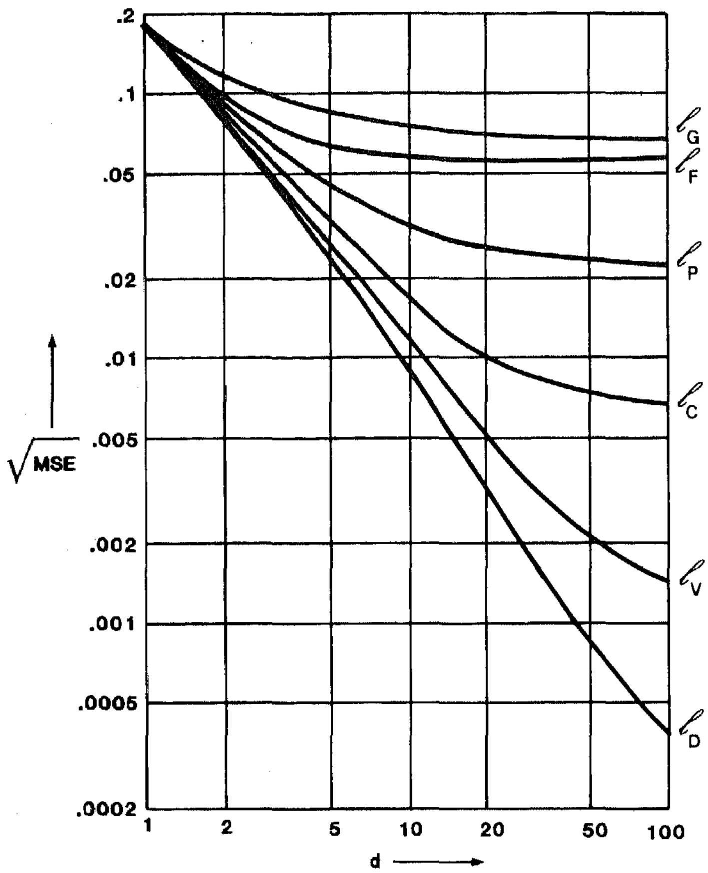

figure 3. A plot of the root MSE of the various length estimators as a function of the sampling density d.

The comparisons were done for the following situation. Consider a unit square grid with Cartesian coordinates x and y. An infinite line $y = \alpha x + e$ is digitized. Due to the symmetry of the grid one can assume $0 \leq e < 1$ and $0 \leq \alpha \leq 1$. The digitization method is an 8-connected chaincode, and leads to a straight string consisting of chain codes 0 and/or 1. Consider n codes of the infinitely long string, corresponding to n 'columns' (in the sense defined in [7]) of the grid. It is known how long the continuous line is between these columns, namely $\ell = n\sqrt{1 + \alpha^2}$. The estimated length of the string is calculated by one of the methods described and yields estimate ℓ_E. The square error $(\ell_E - \ell)^2$ is then averaged over all lines. Each line is weighted by its probability of occurrence, assuming a uniform distribution in the polar parameters r and ϕ [3].

This procedure need not be performed as a Monte-Carlo-simulation. Using the formulas for the domain of a chaincode string (a domain D_c of a string c is the set of all lines that could have generated c) given in [2], one can perform the calculation of the MSE exactly. The calculation was done for all estimators, and for various string lengths n=1,2,5,10,20,50,100. The root MSE was normalized per chaincode, to make it a dimensionless quantity.

RESULTS

Figure 3 shows the result of the evaluation for the different estimators. It is seen that for short strings all estimators behave similarly: the MSE is high, and approximately equal. For very long strings the estimators show remarkable differences: some MSE's reach limit values. The Freeman estimator, used universally, will never become more accurate than 6.6%. This is partially caused by its bias, but when this is removed (resulting in ℓ_P) the inaccuracy is still 2.6%. The corner count method is the best of the methods using a linear formula; it reaches 0.6% in the limit. The optimal estimator ℓ_D will never reach a constant limit value. From the figure it is empirically found that its limit is

$$\sqrt{MSE} = .3\ d^{-1.5} \qquad (11)$$

where d is the sampling density, the number of columns per unit length. This is the best accuracy that can be achieved for length measurement.

It may be slightly unexpected that the linear methods do not become more accurate even at infinite sampling densities. The explanation is found from figure 2: these methods will always fit an octagon to a circle. Even if this is done optimally, as Proffitt and Rosen did with ℓ_P, there is still a residual error.

From the figure one can see what estimators are suited for straight lines according to their accuracy. In the next section we will discuss the algorithmic complexity.

COMPUTATIONAL ASPECTS

Usually one is interested in the length of an arbitrary curve rather than the length of a straight line. As far as we know, no estimators have been developed especially tailored even to specific curves such as circles or ellipses. Therefore, the best one can do at present is to apply the estimators for the length of a straight line to the chaincode strings of curves. It is not known yet how accurate these estimates are, but presumably the MSE of the estimators and their ranking are similar to the case of straight line length measurements. (Comparative measurements on the performance of the various estimators on circles are being performed. We hope to report on these experiments in the near future).

The application of the first three methods to arbitrary chaincode strings is straightforward: one just counts the number of odd and even codes in the chaincode string of the contour, and uses the estimation formula. The corner count method requires the evaluation of n_c, but this is also straight-forward. The formulas to be evaluated are linear, and this makes them very fast to compute.

The two methods ℓ_V and ℓ_D are much more complicated, and intimately connected to straight lines. If used for arbitrary curves they require a piece-wise-linear decomposition to be made (this can be done in O(N) time, if N is the number of codes in the contour), and the calculation of the parameters (n,m,k) or (n,q,p,s). This also requires O(N) time. Then the formulas (8) or (9) should be evaluated. These formulas contain logarithmic and exponential functions, requiring complicated floating-point arithmetic. A table containing the functions f and g will improve the speed of calculation.

CONCLUSIONS AND RECOMMENDATIONS

The conclusions that can be drawn certainly apply to the measurement of the length of a straight line. As mentioned before, work is being done to test the validity for arbitrary contours.

- The estimation ℓ_F and ℓ_G are out of date. With equal ease one can reach better results using ℓ_P, which is 2.5 times more accurate.
- Only in time-critical situations should one use ℓ_P. The corner count method ℓ_C is much more accurate (up to .6%) than ℓ_P, and hardly requires more computational effort.
- The sub-optimal estimator ℓ_V is outdated. With equal effort one can compute the optimal estimator ℓ_D.
- The optimal (BLUE) estimator ℓ_D is mainly of theoretical interest, since it comprises the best accuracy to be achieved in length measurement on digitized data. It's practical use is probably limited to situations where high accuracy is required, but (over-)sampling is expensive.

Summarizing, the corner count estimator (7), treated in [6] and [7] is the estimator to use in most practical situations. It is easy to compute, both for straight strings and for arbitrary contours. And it is reasonably accurate, ranging from 3% for strings of length 5 via 1.5% for length 10 to a limit accuracy of .6%.

ACKNOWLEDGEMENT
This research was partially supported by Stichting FOM, The Netherlands, under project DTN93.0329.

REFERENCES

[1] Borgefors, G., Distance Transformations in Arbitrary Dimensions, Computer Vision, Graphics, and Image Processing, 27, (1984) 321-345.

[2] Dorst, L. and Smeulders, A.W.M., Discrete Representation of Straight Lines, Pattern Analysis and Machine Intelligence, vol. PAMI-6, vol.4, (July 1984) 450-463.

[3] Dorst,L. and Smeulders, A.W.M., Best Linear Unbiased Estimators for Properties of Digitized Straight Lines, internal report, Delft University of Technology, Delft, The Netherlands (1985).
A short digest of this work can be found in :
Dorst, L. and Smeulders, A.W.M., The Estimation of Parameters of Digitized Straight Line Segments, Proceedings of the 6th International Conference on Pattern Recognition, Munich, Oct.19-22 (1982) 601-603.

[4] Freeman, H., Boundary Encoding and Processing, in Picture Processing and Psychopictorics (B.S. Lipkin and A. Rosenfeld, eds.), Academic Press, New York (1970) 241-266.

[5] Groen, F.C.A., and Verbeek, P.W., Freeman Code Probabilities of Object Boundary Quantized Contours, Computer Graphics and Image Processing, 7 (1978) 391-402.

[6] Proffitt, D., and Rosen, D., 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.

[7] Vossepoel, A.M. and Smeulders, A.W.M., Vector Code Probability and Metrication Error in the Representation of Straight Lines of Finite Length, Computer Graphics and Image Processing, vol.20, (1982) 347-364.

DISCUSSION

Van Heel:
Have you ever tried to apply your method to real problems like fractional coast lines?

Smeulders:
No, you should keep in mind that our procedure is for straight lines and not for arbitrarily curved ones. Any other line than a straight one will give a biased result. One can in general not solve it in a model when you know nothing about your object. It is our experience that for reasonable sampling circumstances the approximation of a straight line is reasonable. Actually, only one or two percent of error is added.

Haralick:
What would happen if you did just a least squares fit to the points of the line and estimate the length from the coefficients of the estimated line?

Smeulders:
That is an alternative approach and you can also think of splines. Personally, I believe that filtering the contour with a differentiating filter in order to find the curvature and base your estimate on that curvature is one of the most accurate computational methods available.
Also spline functions give a very accurate result, even more accurate than the linear ones. For practical applications the filtering is the best approach since it removes noise. The BLUE method does not do that, at least not for the moment.

Young:
What is the computational complexity of the procedure?

Van Otterloo:
The complexity of the least square fit is proportional to the number of points you have in the data window.

Smeulders:
The BLUE estimator can only be used by tabulating the results for every possible combination. But when you look at the first 20 possible codes, then you already run out of computer memory.
We have also done Taylor series expansions to these BLUE estimators and by taking a first or second order approximation one has a next to optimal solution.

Choudry:
There are several problems here. Just a couple of minor points. First of all you started to say you did contour analysis. This method is strictly for line segments. There is no analytical way to deal with for example circles. Your procedure is based on the e-alpha plane and I do not see that there is an easy way to do that for more complex lines.

Smeulders:
I even think that it is impossible to parameterize a circle by a finite number of variables to make an estimation of the true length.

Stockman:

I just want to point out that around 1980 a paper appeared in Pattern Recognition dealing with a simulation study on arbitrarily shaped blobs. They coded the boundaries and used 5 different estimators. The errors were about 6% and none of the algorithms was better than the others.