Science on Form: Proceedings of the First International Symposium for Science on Form,
General Editor: S. Ishizaka, Editors: Y. Kato, R. Takaki, and J. Toriwaki, 421-430.

Extraction of Lines of Unspecified Texture from Unconstrained Line Drawings

T. Kasvand

National Research Council Canada, Division of Electrical Engineering, Ottawa, Ont., Canada, K1A 0R8

Keywords: Line Texture, Segmentation, Drawing, Chart, Map

Engineering drawings, maps, patterns, and similar material frequently contain mixtures of lines of different textures, such as dotted lines, dashed lines, and variously marked lines. The textures are used to facilitate human vision. Methods are proposed for segmenting images containing mixtures of textured lines. For generality no a priori information is assumed to be available neither on the texture elements nor on the shapes of the lines. The processing consists of two major steps. First, the basic elements are extracted, regularized, described, and clustered. Then the selected (sets) of labelled clusters are projected back into the image space and converted to solid lines. This allows the basic strategy in the first step to be repeated at a higher level of abstraction.

INTRODUCTION

An apparently novel problem is the computer analysis of (linear) textures which appear as curves in two- or three-dimensional space. Linear textures are less frequent in nature but are common on maps and graphs, in engineering drawings, dress patterns, and so on, where it is standard practice to use lines of different structure in order to make it easier for the human eye to distinguish the lines. For us these lines "stand out" immediately, even though no specific a priori information has been given on the structure of the lines, see Figure 1. On engineering drawings and maps certain conventions have been accepted, but in general there is no precisely defined a priori information on the lines or on the texture and structure of the elements out of which the lines are composed. Possibly the most developed representations are found on certain dress patterns to indicate various shapes to be cut. The user of these patterns is expected to have normal vision, but is not trained in a technical sense.

Textures are normally associated with surfaces where they appear as more or less regular but differing sets of repeated features covering the areas. Some approaches may be found in Gagalowicz and Ma, Matsuyama et al., Therrien, Toriwaki et al. Such techniques could only be applied along each curve, but the locations of the curves are not known. Likewise, if no assumptions are made about line shapes, the model based techniques (Ballard, Yamada) become inapplicable.

A sequence of processing steps is suggested in order to extract the various textured lines from an image in the absence of a priori information on the texture. However, the practical applications may be remote, since as yet we are not too concerned with the design of generalized computer vision systems. In a specific practical application there are always shortcuts indicated by a priori knowledge which would make many of the suggested procedures superfluous.

PROBLEM DESCRIPTION AND DEFINITION

From the illustrations in Figure 1 it is immediately apparent that the problem of extracting individual textured curves from such images has several levels of complexity which are largely dependent upon the amount and type of a priori information we have or are willing to use in the solution. An interesting visual characteristic of the textured curves is that the more elements they have the easier it is to see the curves. From the computational point of view the opposite occurs if combinatorial methods are contemplated.

The images may contain coloured lines, solid lines of differing thickness, solid wiggly lines, parallel lines, dashed lines, dotted lines, and lines composed of various marks and sequences of marks, or any combinations of them. Several distinct but criss-crossing curves may be composed of the same elements. Texture elements may touch or cross continuous curves and form "cross-hatchings", and the texture elements vary from simple straight line segments to rather complicated shapes and sequences of repetitive sets of elements. For example, a simple textured curve may consist of a string of ....., or ///////, or CCCCCC, while a set could be a repetitive sequence like /C/C/C/C/C/C/, CC..CC..CC.., etc. Basically, there is no limit to the possible number of combinations that could be invented. The texture elements of the curves do not have any predefined shape or size. Orientation of the elements may or may not be important. In the case of a space curve, the thickness, size, and shape of the texture elements also vary systematically with distance and the view angle between the observer and the direction of the space curve. In the two-dimensional case the texture elements are essentially of the same shape and the elements are spaced approximately equally along the curve.

It will be assumed that there is no a priori information available on the contents of the input image, except that it is an image of two-dimensional line structures rather than an image of any scene. The initial steps in processing, such as scanning and colour separation, are quite obvious. Clearly, the directly measurable pixel features, such as colour, and results from local processes, such as local line thickness, are primary features for curve segmentation. However, the intrinsically interesting problems occur when all the lines in a binary image are of equal thickness and all the pixel based cues, such as colour, have been removed or ignored. A symbolic example of such an image is shown in Figure 2. Only this problem is discussed in the rest of the paper.

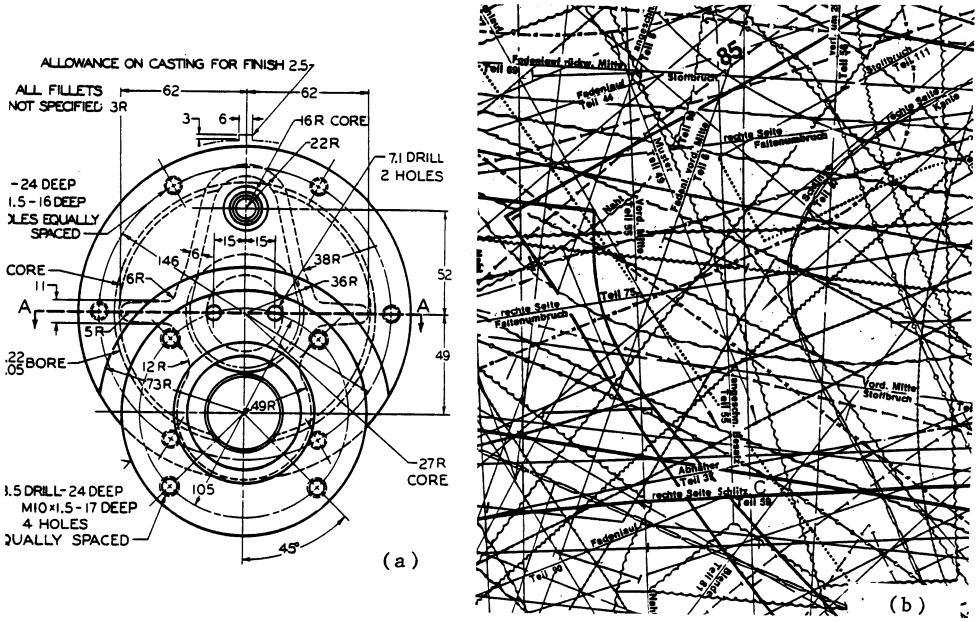

Figure 1. Examples of textured lines in practice. (a) A portion of an engineering drawing. If a pencil is used then the lines are of approximately equal thickness. (b) A portion of a dress pattern. The lines on the original are in colour.

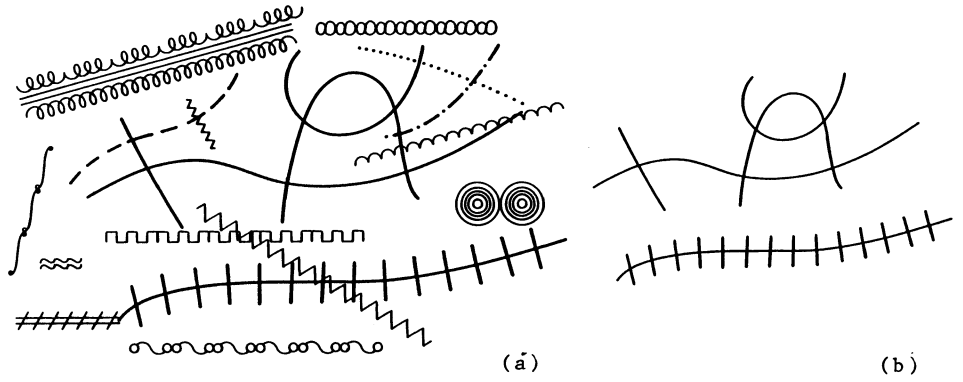

Figure 2. Conceptual examples of textured curves. (a) A collection of textured curves. Even though no a priori information is available on either the shapes of the lines or on the textures (the "blind" case) we immediately see the curves as distinct. (b) A subset of (a) for illustrating the processing steps. The lines are of equal thickness.

LINEAR TEXTURES IN TWO DIMENSIONS

A binary image of line structures is given, for example as shown in Figure 2. The lines are assumed to be of approximately equal thickness and allow "perfect" thinning (Tamura). Nothing else is known. Thus, the textured curves in the binary image consist of an unknown number of texture elements, the curves are of undefined shape, the number of curves and their lengths and orientations are unknown, and the curves may criss-cross. The only characterizing features of such images are that the curves are either composed of repetitive sets of sequences of similar elements and/or the curves have some other unspecified texture characteristics which make them differ from the other curves in the image. Consequently, the texture element is by definition unknown both in size and in shape, and the importance of its orientation is uncertain. Given these premises, a rather general processing strategy is required, which in the present case consists of the following main steps.

a) Thin the lines, detect junction or crossing points, correct thinning errors, establish a pixel sequence on each line segment, detect critical points, and label the junctions and the segments uniquely. The results are illustrated schematically in Figure 3 for some lines. Each segment Sk , k=1,2,..., (Sk = portion of line between junctions and/or free ends) may now be approximated by a continous piece-wise linear function. Enough information can be extracted from the processed images to represent the problem as a graph, to apply syntactic methods (You and Fu), or to apply expert systems (Nazif and Levine). In practical cases there may be now enough information to apply task-specific techniques to solve the problem. However, the preprocessing may also be continued in the absence of prior information.

b) Establish good connectivity across junctions, label the resulting lines, and establish a pixel sequence along each line. Reduce the pixels on each line to a connected set of piece-wise linear vectors, see Figure 4. Regularize the vector strings (Kasvand and Otsu, 1983) to obtain a normal coordinate description C(s) for each line. C(s) represents curvature versus distance s along the line. The resultant labelled lines will be called "elemental components" E of the textured curves. The characteristics of each elemental component (Ek, k=1,2,...,n) in the image are now available in a computationally convenient form. The shape of each line is given by the Ck(s) function while the position (Xk,Yk), orientation Ak, and length Lk of the line are given as initial conditions. Normalization of C(s) for length L of the line is not needed if only one image is analysed, unless scale independence is required. This may occur within a single image if, for example, the elemental components C and c in curves such as CCCCC, ccccc, CcCcCc are to be considered equivalent, or if several images with unknown scales are used. This may be another convenient place for task-specific solutions, such as recognition of complex characters.

The "good connectivity" rule states that "at places where segments meet or cross the crossing segments are joined by assuming that the lines are 'as straight as possible'" (Kasvand and

Otsu 1984a). The good connectivity rule is generally valid since it corresponds to one of the many ways that the human visual system tends to organize the lines into larger units. However, there are certain exceptions which require specific rules. On a machine drawing, for example, the dimensions are indicated by dimension lines and arrows which terminate the lines. The arrow heads frequently meet "head on". If the connectivity rule "knows nothing" about arrow heads, the dimension lines will simply be joined across the arrow heads.

c) In addition to the Ck(s) function, construct an associated description Bk(s) for each Ek (elemental component) which gives the position of the crossing or junction points Jj as a function of distance (s) along Ek. More informations should be tabulated, such as the positions of the crossings (si), the label of the crossed junction Jj, the crossing angles Ba at each junction Jj between the two elemental components Ek and Em (angle (Ek(si),Em)), the label of the crossed elemental component Em, the relative position sj of Jj on the crossed elemental component Em, etc. This gives a list of the form Tk(si,Jj,Em,Ba,sj,...), k = 1,2,...,n, i = 1,2,..., etc., where n is the number of elemental components in the image.

d) Construct a "proximity table" D(Ei,Ej) which gives the closest distance between any two neighbouring elemental components Ei and Ej, i = 1,2,...,n, j = 1,2,...,n. The D(Ei,Ej) values are easily obtained from an image where each pixel is labelled with its corresponding E value. Spread or dilatate the labels and record the iteration count for the "first encounters" between the labels. The E's that cross have distance of zero. The procedure is illustrated symbolically in Figure 5. The proximity table D(Ei,Ej) could be extended to include second encounters (first encounters across the elemental components), etc. However, it is useful to also construct a coincidence count table Dc(Ei,Ej,m) where m = 0,1,2,... . Dc(Ei,Ej,m) gives the number of pixels where the labels Ei and Ej meet during iteration cycle m. Clearly, if Ei and Ej are in parallel, then the pixel count (as a function of m) has a large peak value, which is close to the length of the shortest of Ei or Ej (if length is measured in pixel count).

The preliminary computations are quite lengthy, but the processing requires no a priori information on the contents of the image. The results so far have been obtained from a set of "bottom up" procedures, or the results are entirely "data driven". The accuracy depends on how well the lines can be thinned and on the performance of the good connectivity rule. Experiments have indicated that many thinning errors can be corrected and that the good connectivity rule can be prevented from joining segments in dubious cases. Due to space limitations, the details could not be illustrated with actual examples. Again, in particular applications where the texture elements are known a priori, this information should be more than ample for the detection and recognition of specific textured curves. The challenge, as before, is what to do if the a priori information is absent.

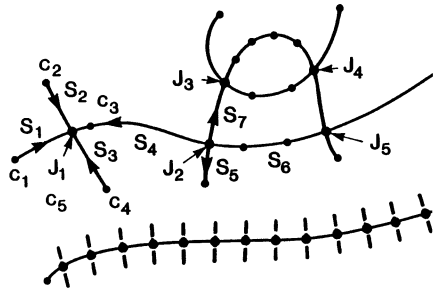

Figure 3. Results from preprocessing before use of the good continuity rule. J1,J2,... = labelled junction or crossing points. S1,S2,... = labelled segments (line sections between junctions and/or free ends). J1,J2,... and c1,c2,... = critical or break points on segments for continuous piece-wise linear approximation. --> (arrow) = pixel sequence along segments. The direction is arbitrary.

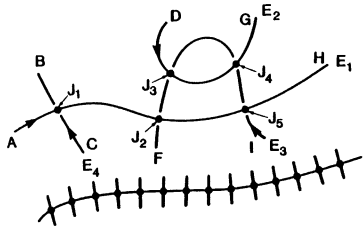

Figure 4. Results after applying the good continuity rule to the curves in Fig. 3. The connected lines are marked with letters, where, elemental component E1 = line A to H, component E2 = D to G, E3 = I to F, etc. The labelling sequence (k) and the direction (-->)of the elemental components Ek, k = 1,2,...,n, is arbitrary. The critical points (and junctions) are the same as in Fig. 3.

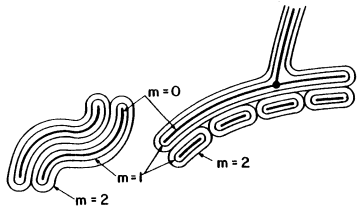

Figure 5. A symbolic illustration of the dilatation process. The iteration cycles are indicated by m, where m=0 for original labelled pixels, m=1 for the first dilatation, etc.

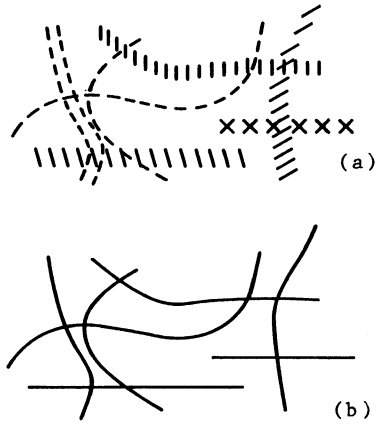

Figure 6. A conceptual example of the results of back projecting a labelled cluster for an elemental component representing a short line segment (/) of arbitrary orientation. (a) The back projected image. (b) The elements in (a) have been joined with continuous lines. The image is now conceptually identical to, for example, Fig. 2(b), i.e., it is a structure of lines which can be processed by using the initial preprocessing techniques.

SUBSEQUENT PROCESSING STRATEGY

Continuing with the present "blind" study, the question is, What is the higher level strategy in the present situation? Clearly, whatever the rules, they can only be meaningful if defined in the context of the presently available information.

Studies of the various textured lines indicate that in general there are two types of curves, namely, the long continuous curves with or without various forms of "crosshatching", and the long broken curves consisting of isolated texture elements. In principle there is no upper limit to the complexity of a long curve neither on its shape nor on the complexity of the texture elements. A long curve can twist and turn in many ways and cross itself and other curves frequently. A shorter curve is less complex since minimally about five texture elements are needed to define a curve. The texture elements themselves vary from simple isolated elemental components, such as a dot (.) or a dash (-), to sets of elemental components of varying complexity (././././, xxxxx). Normally, the texture elements cannot be too far from each other, or the curve "falls apart" visually. Exceptions usually consist of solid curves that are "ticked off" with an occational mark or symbol to simplify visual separation at dubious crossing points.

The longer continuous curves could have periodic structure which is detectable from the analysis of their C(s) and B(s) functions. Fourier analysis is expected to be suitable. The crosshatched long curves may be separated from the others by the frequency of the crossing points given in the B(s) function. All the elemental components that cross a given long continous curve Ek are obtainable from the Tk(.,.,...) table, and may be subjected to separate analysis. Parallel long curves have a large encounter frequency per length, obtained from the Dc(.,.,m) table, but some counter examples can be constructed. There is also the possibility that long continous curves have been left broken by the good continuity rule. Subsequent procedures for long continous curves are analogous to those for long curves consisting of separate texture elements, see below.

Measures for the shapes of the short elemental components Ei (a subset of all Ek, k = 1,2,...n) can be computed (Kasvand and Otsu 1984b). However, in the absence of a priori knowledge of the possible shapes in the image, the measures or features Fif, f = 1,2,..., should not be very elaborate. Only position and orientation invariant features should be used in the general case, since as yet there is insufficient knowledge whether, for example, a dashed curve exists where the dashes may be orientated along the slope of the curve. The length of the elemental component, its absolute average curvature, variation of curvature, absolute value of skewness of curvature, and closure should be sufficient initially. (Closure = distance between end points divided by total length of curve, for example.) The number of crossing points from the Bi(s) function and whether the elemental component crosses itself are also generally useful.

The higher level strategy that suggests itself consists of the following steps.

1) Separate the long and short elemental components. Long and short components require different approaches. Histogramming based on the lengths of the elemental components should show clearly definable peaks, even though some counter-examples could be contrieved. The subsequent steps are intended for the short components.
2) Use of proximity to select the elemental components that are close to each other in the image space. The first encounter distance D(Ei,Ej) between any two elemental components Ei and Ej is available from the proximity table. Only those elemental components that are close to each other can form continous curves.
3) Use of similarity to group elemental components of the same kind. However, the prevailing shapes in the image are unknown. One thus has the problem of finding which features cluster or "cluster whatever can be clustered". The use of too many features may scatter or fragment the textured curves, too few do not separate the elemental components sufficiently.
4) On the assumption that clusters have been found in step (3), label the clusters, and project the cluster labels back into the image space, i.e., create an images of the cluster labels to which each pixel belongs. Figure 6a illustrates symbolically an image obtained by back projecting the cluster corresponding to short line elements (/), irrespective of orientation.
5) Compute the proximity table L(Li,Lj) between the cluster labels Li and Lj, i = 1,2,..., j = 1,2,..., in the back projected image. The procedure is identical to the one used for obtaining the proximity table Dc(Ek,Ej,m) for the segment labels.
6) The frequently occurring pairs of cluster labels (Lm,Ln) which are close to each other are very likely to represent parts of the sets of elemental components that form complex texture elements. For example, a curve composed of /c/c/c/c/c has the texture element /c, where, say, Lm represents the cluster for /, and Ln is the cluster for c.
7) Give the frequently occurring proximal cluster pairs new unique labels in the back projected image. For example, (Lm,Ln) becomes Lq.
8) Iterate steps (5) to (7) until there are no more frequently occurring proximal label pairs in the back projected image. Each different complex texture elements should now have one unique label.
9) Create a separate image for each label, for example, as in Figure 6a for the unorientated short line element (/).
10) Since the elemental components for each complex texture element is now known, join the elements with solid lines, for example, as shown in Figure 6b. Thus, a curve of unknown texture has been converted to a solid line. The segmentation of these lines is identical to the steps in preliminary processing.

At the present moment the necessary steps for preprocessing have been completed and the cluster are labelled, but the proximity analysis and the iterative steps are yet to be programmed. Methods based on minimum spanning trees have been used to study

feature clustering. The really interesting problems occur when the images contain large numbers of many different texture elements. Such images, however, are large and require considerable computer resources.

CONCLUSIONS

A rather elaborate sequence of processing steps has been proposed to segment two-dimensional images containing arbitrarily textured curves. It is a "blind" study, where no a priori information is used about the structure of the texture elements nor have any constraints been imposed on the image content, except that the image should contain arbitrarily textured curves. A purely bottom-up or data driven approach has been used to extract the individual textured curves. The larger the image and the more texture elements there are in the image, the better the results. This corresponds to a general observation about human vision that the more texture elements there are on a curve, the easier it is to see it, and there is no need to know the texture of the curve in advance. The most interesting and also the most complex case occurs when all the textured lines in the image are of equal thickness and have the same gray level values. The combinatorial explosion has been avoided but the method is still very computation intensive. Most of the computations are devoted to preprocessing for which suitable hardware can be designed. If a priori information is permitted then many of the processing steps can be bypassed.

ACKNOWLEDGEMENTS

The author is most grateful to the Division of Electrical Engineering, National Research Council Canada, where these studies are being carried out.

REFERENCES

Ballard, D.H., Generalizing the Hough Transform to Detect Arbitrary Shapes, Pattern Recognition, Vol. 13, No. 2, 1981, pp. 111-122.

Gagalowicz, A., and Ma, S.D., Synthesis of Natural Textures on 3-D Surfaces, Proc. 6'th IJCPR, Munich, Oct. 19-22, 1982, Vol. 2, pp. 1209-12,

Matsuyama, T., Miura, S., and Nagao, M., Structural Analysis of Natural Textures by Fourier Transformation, Computer Vision, Graphics, and Image Processing, 24, 1983, pp. 347-362.

Kasvand, T., and Otsu, N., 1983, Regularization of Digitized Plane Curves for Shape Analysis and Recognition, SPIE, The International Society for Optical Engineering, 27th Annual International Technical Symposium & Instrument Display, Conference 435, Architecture and Algorithms for Digital Image Processing, SPIE Proc. vol. 435, San Diego, California, USA, Aug. 25-26, 1983, pp. 44-52.

Kasvand, T., and Otsu, N., 1984a, Segmentation of Thinned Binary Scenes with Good Connectivity Algorithms, Proceedings of the 7th International Conference on Pattern Recognition, Montreal, Que., Canada, July 30 - Aug. 2, 1984, pp. 297-300.

Kasvand, T., and Otsu, N., 1984b, Recognition of Line Shapes Based on Thinning, Segmentation with Good Connectivity Algorithms, and Regularization, Proceedings of the 7th International Conference on Pattern Recognition, Montreal, Que., Canada, July 30 - Aug. 2, 1984, pp. 497-500.

Nazif, A. M., and Levine, M. D., Low Level Image Segmentation, An Expert System, PAMI-6, No. 5, Sept. 84, pp. 555-577.

Tamura, H., A Comparison of Line Thinning Algorithms from Digital Geometry Viewpoint, IJCPR, Kyoto, 1978, p. 715-719.

Therrien, C.W., Linear Filtering Models for Texture Classification and Segmentation, Proc. 5'th IJCPR, Miami Beach, Dec. 1-4, 1980, pp. 1132-5.

Toriwaki, J.I., Yashima, Y., and Yokoi, S., Adjacency Graphs on a Digitized Figure Set and their Applications to Texture Analysis, Proc. 6'th IJCPR, Munich, Oct. 19-22, 1982, Vol. 2, pp. 1216-8.

Yamada, H., Contour DP Matching Method and its Application to Handprinted Chinese Character Recognition, Proc. 7'th IJCPR, July 30 - Aug. 2, 1984, Montreal, Vol. 1, pp. 389-392.

You, K.C., and Fu, K.S., A Syntactic Approach to Shape Recognition Using Attributed Grammars, IEEE Trans., SMC-9, No. 6, June 1979, pp. 334-345.

10-4
Q: How about the possibility to apply the Hough transformation technique to these kinds of pictures? (J. Toriwaki)

A: Hough transform is applicable only if the shape of the line to be found in the image (model) is known in advance. The emphasis in the paper was on unconstrained line structures, which means that there is no model. The problem is to find the lines without the need of any a priori knowledge.