SUBSAMPLING: A STRUCTURAL APPROACH TO TECHNICAL DOCUMENT VECTORIZATION

PASCAL VAXIVIÈRE and KARL TOMBRE
CRIN/CNRS & INRIA Lorraine, Bâtiment LORIA
B.P. 239, 54506 Vandœuvre CEDEX, France
Karl.Tombre@loria.fr

ABSTRACT

Vectorization, i.e. raster-to-vector conversion, is a fundamental step in technical document image analysis. Many methods have been proposed for performing this task. In this paper, we present a method which extends and improves a subsampling approach first proposed by Lin, Shimotsuji, Minoh and Sakai, which is based on the decomposition of the image into *meshes*, the lines being recognized by analyzing their intersection with these meshes. Our improvements to the original method include overlay meshes, mesh simplification, line following, localization and thickness computation, and a set of corrections to improve the final result.

1. Introduction

A fundamental step in technical document image analysis is that of *vectorization*, i.e. the conversion of the graphics parts into vectors. This is also often called the raster-to-vector conversion. Many methods have been proposed for performing this, most of them being based on some kind of skeletonizing by thinning or distance transform, followed by polygonal approximation[11,10,16]. Commercial systems usually integrate this kind of approach[22]. It is also usual to include specific methods for recognition of circular arcs[9], dashed, dotted or dot-dashed lines[12], or even more complex graphic features such as hatched areas.

But whereas skeletonizing is a typical digital image processing approach to vectorization, based on topological or metric methods, numerous other techniques have been proposed, taking a more structural approach. These include, among other methods:

- the *line sensor* method which follows directly a line by keeping a tracking window perpendicular to the line direction[7];
- the methods based on the analysis of the *line adjacency graph*[3,15,17];
- Dov Dori's *Orthogonal Zig-Zag* method (OZZ), which intercepts a line and then zigzags inside this line in order to find its main direction[4,6];

- the *maximum squares* method, which is a kind of subsampling of the image in terms of maximum black squares, the line being extracted by joining the centers of aligned maximum squares[18];
- various methods based on the parallel following of the opposite contours of lines[1], the matching of opposite segments resulting from the polygonal approximation of these contours[5,14,2,19], or the combination of this kind of contour matching with a skeleton computed on the image[8];
- a typical subsampling approach proposed by Lin, Shimotsuji, Minoh and Sakai and based on the decomposition of the image into *meshes*, the lines being recognized by analyzing their intersection with these meshes[13]. This approach is the starting point of the method we propose in this paper.

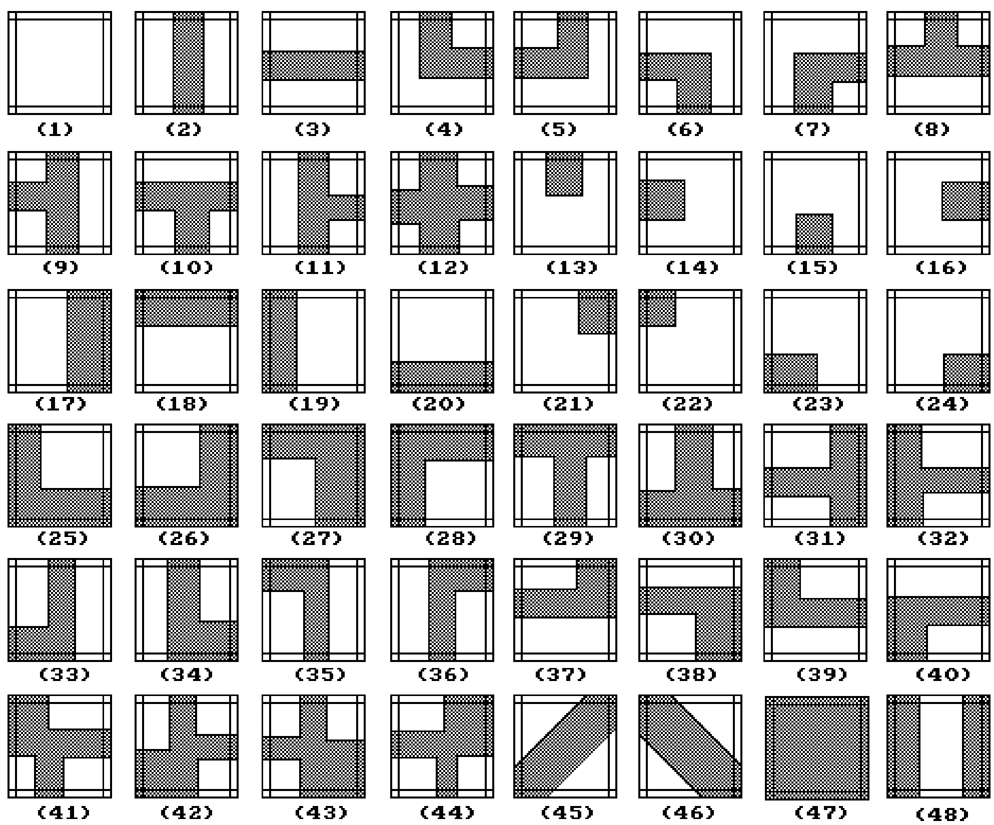

Figure 1: The original mesh configurations, as proposed by Lin *et al.*

In the next section, we will summarize Lin *et al.*'s method; then, we will describe how we improved the method in various ways, first in the CELESSTIN system[21], but also further on, as the method has continued to be improved this last year.

2. The mesh method

The basic idea in the mesh method is to split up the image into meshes of size $n \times n$ pixels, with a value n chosen so that any mesh should *a priori* only intersect with one line. This is the basic idea of subsampling applied to vectorization: by looking only at the intersection of the mesh sides with the binary image, we make hypotheses about the local line configurations. 48 characteristic meshes can be determined, as illustrated by Fig. 1. A set of structural rules then lead to the extraction of the line vectors from the observed local context of each mesh.

In our system, we represent each mesh by looking separately at each of its sides, as illustrated by Fig. 2. A special code is given to meshes which do not correspond to

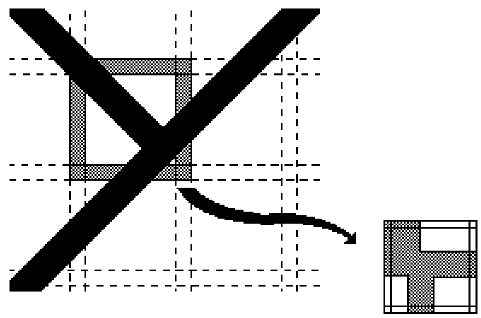

Figure 2: Coding of the mesh.

one of the characteristic meshes of Fig. 1. In the original article[13], such unrecognized meshes were analyzed by some local symbol recognition method. We improved this in Celesstin II by decomposing recursively the unknown mesh[20]. But it was not sufficient for all possible cases and made line following difficult. Hence, we continued to enhance the method, as we will explain in the next section.

3. Our enhanced mesh method

3.1. Overlay meshes

The two important pieces of information which must be extracted from this coding in terms of meshes are the main lines in the drawing and the junction and extrema points. This requirement guides all the vectorization procedure.

Instead of splitting up unrecognized meshes, we adapted a well-known notion from Cad, namely superimposed layers. Each mesh which is too complex to be coded according to the original scheme is decomposed into several superimposed meshes with simpler codes, as illustrated by the four cases of Fig. 3. The decomposition method is based on the first requirement, that of finding the main lines. Hence, an unrecognized mesh is first split up into a mesh keeping the sides which are completely black in the original mesh and an overlay mesh containing the complement. The latter

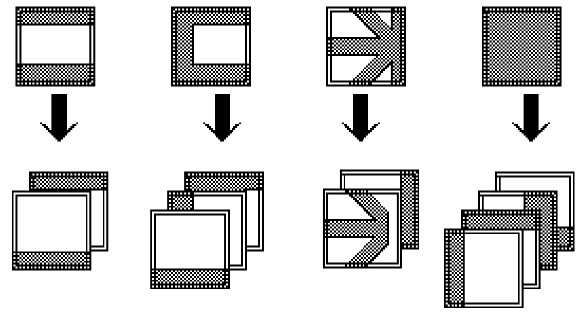

Figure 3: Four cases where a mesh is decomposed into overlay meshes.

is further recursively split up into overlay meshes until each of them corresponds to a know configuration. Fig. 4 illustrates this two-step splitting method.

3.2. Mesh simplification

This splitting procedure stills leaves us unsatisfied in some cases: the mesh configurations represented by Fig. 5 correspond to lines running along the border between two adjacent meshes. In this case, it is desirable to merge these two meshes into a single one, as illustrated by Fig. 6. The result is a structural description of the original image as a set of meshes, some being single and others superimposed layers, with only a limited set of possible configurations (Fig. 7), all very simple to take into account in the next step, line following.

3.3. Line following

The next step is to follow the lines from the "mesh maps" generated by the previous phases. We now have a limited set of mesh configurations, so the complexity of the following procedure is reduced. As said previously, the two important elements to extract are line vectors and junctions.

3.3.1. Types of vectors

Different kinds of specific vectors can be determined at this phase:

- horizontal and vertical vectors: we assume that the horizontal and vertical directions in the drawing are preponderant, so it is important to extract them. Following is easy in this case, as it just consists of chaining successive meshes vertically or horizontally, when their codes are homogeneous;
- dashed and dot-dashed lines: chains of meshes with regular combinations of "13" and "15", or "16" and "14" mesh codes, correspond to this kind of lines;
- oblique lines: they are mainly the components of hatching. In some cases, their

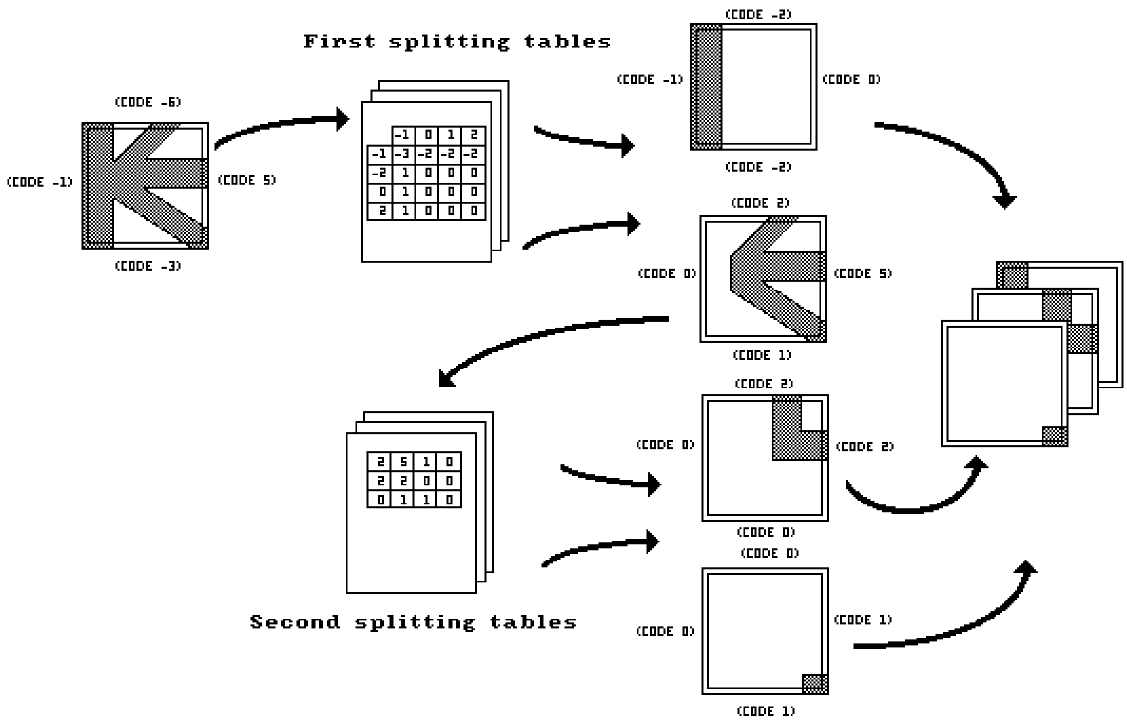

Figure 4: Splitting up an unrecognized mesh into overlay meshes.

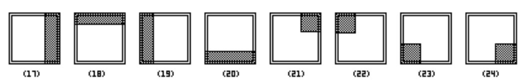

Figure 5: Mesh configurations where two meshes can be merged.

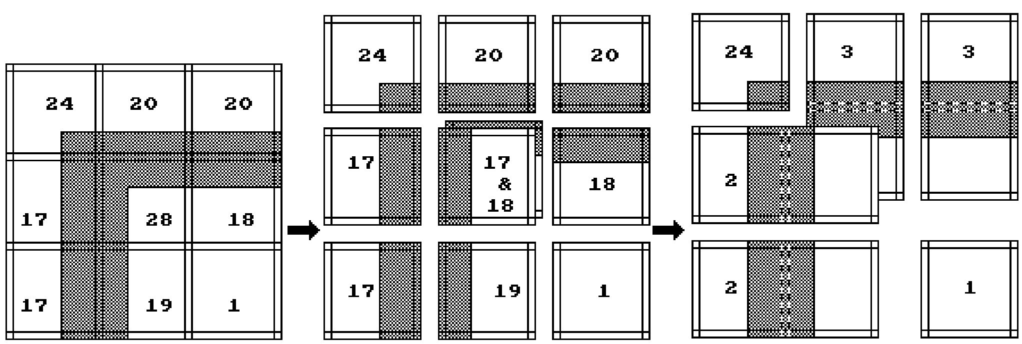

Figure 6: Simplification by mesh merging.

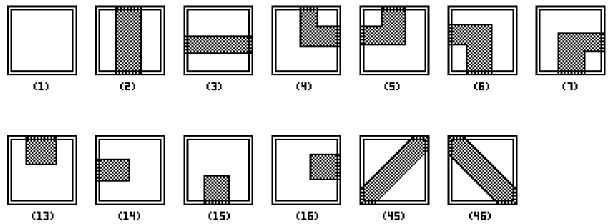

Figure 7: Remaining set of mesh configurations after split and merge.

detection is easy, but in other cases, they will be detected as sets of isolated dashed segments, and a higher-level combination step must be introduced.

3.3.2. Junctions and line extrema

When we split up meshes, we build a table of the line intersections, as we must not lose this information by just following the evident lines. As illustrated by Fig. 8, there are two cases: in the first case, a junction mesh has actually been detected by the

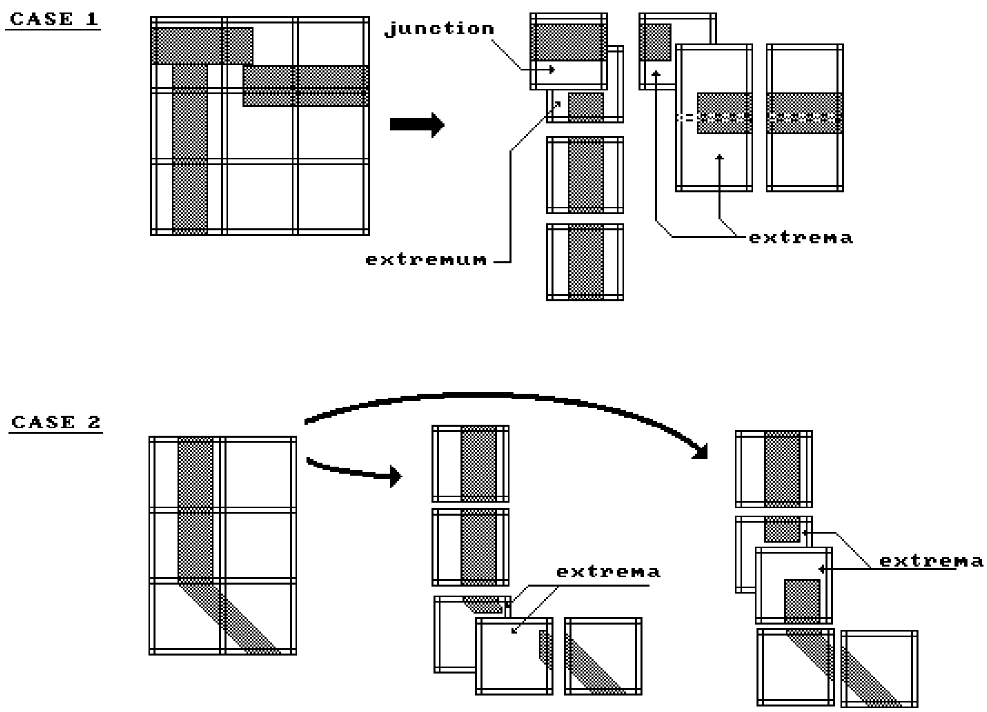

Figure 8: Line junction processing.

subsampling method and this junction is marked as such and is superimposed with the corresponding line extrema. In the second case, no specific junction has been detected but line following detects two line vectors with different directions; depending on the line which is followed first, the location of the superimposed extrema of these two line vectors may vary between two adjacent meshes. This configuration corresponds more to angular points than to actual junctions.

3.4. Precise localization, thickness computation and corrections

Until now, the level of granularity we have worked with is that of the chosen subsampling, i.e. the mesh size. But in order to locate the lines properly and with sufficient precision, we must find the median axis of the extracted lines which have been followed. As the line direction has been found, it is sufficient, once again, to take samples on the raster image, perpendicularly to the line direction, and at positions where we know from the previous steps that there is no junction which might disturb the thickness computation (Fig. 9). We also know which meshes are completely black,

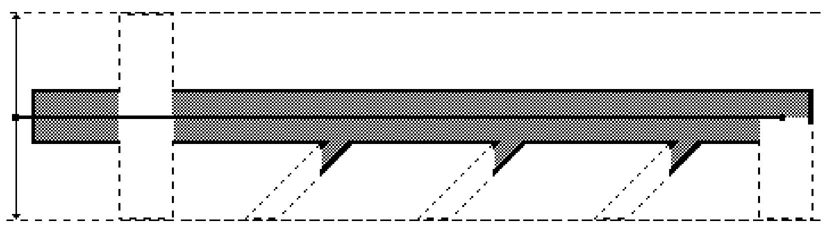

Figure 9: Line localization.

i.e. cover black blobs on the raster image, and we follow the line without taking into account these areas for the thickness and position computation. The result is a positioning and thickness value computed with a pixel precision. The sampling rate for these calculations is to take two samples for each mesh.

Finally, a number of automated corrections are made to the result, to take into account the following cases:

- black blobs: a completely black mesh has been split up into 4 overlay meshes coded 17, 18, 19 and 20, to ensure a correct line following process. But these original black meshes have been marked as such, so that they do not alter the following method and are not taken into account when computing the line thickness, but they are still recognized as such in this last correction step, they are grouped when several adjacent black meshes are present, and the contour of the resulting black blobs is coded in the representation;

- small barbs: small* line segments which are not integrated into dashed lines and are not connected to other segments at their ends are removed;

- merging of small segments: several aligned small segments (case of numerous intersections with hatching lines) are merged into a single line segment.

*In this rule as in the next one, the length threshold for a segment to be considered as "small" is of course scale-dependent and is currently input by the user.

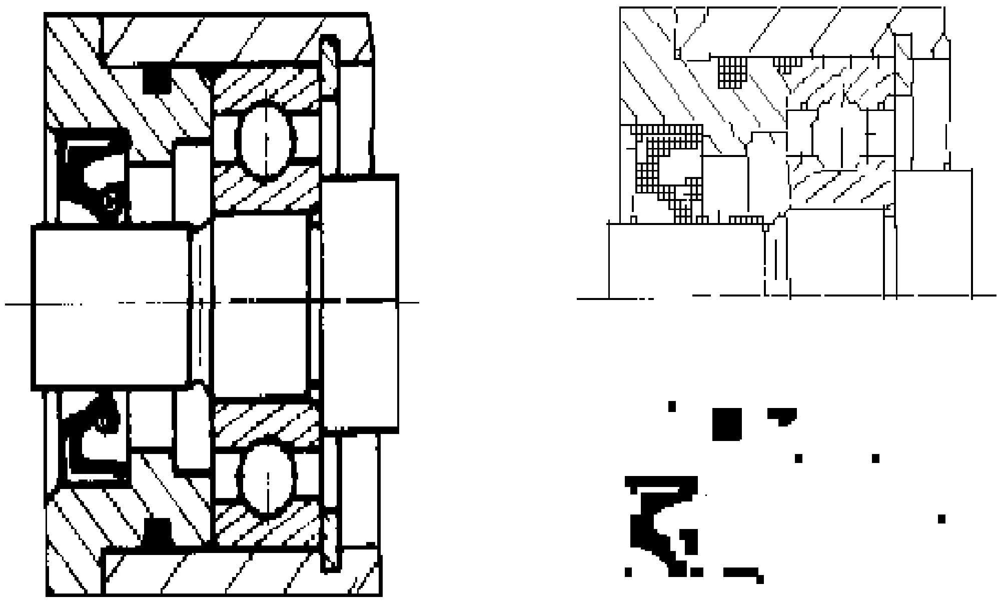

Figure 10: Result of vectorization.

3.5. Results

Figure 10 gives the "brute" result of our vectorization on a drawing. Only the upper part of the vectorization is shown; in the bottom right part of the figure, we show the extracted black blobs. It can be noticed that the contour lines of these blobs are correctly followed on the vectorization without being disturbed by the black meshes, which are represented on the vectorization output as squares corresponding to the associated mesh sides. The junctions are not displayed here, which explains apparent discontinuities in the lines, as each mesh generates an approximate direction for the associated line; but these small segments are actually part of the same line in the associated structure containing the junctions.

4. References

1. K. Abe, Y. Azumatani, M. Mukouda, and S. Suzuki. Discrimination of Symbols, Lines, and Characters in Flowchart Recognition. In *Proceedings of 8th International Conference on Pattern Recognition, Paris (France)*, pages 1071–1074, 1986.
2. D. Antoine, S. Collin, and K. Tombre. Analysis of Technical Documents: The REDRAW System. In H.S. Baird, H. Bunke, and K. Yamamoto, editors, *Structured Document Image Analysis*, pages 385–402. Springer-Verlag, Berlin/Heidelberg, 1992.

3. H. Bley. Segmentation and Preprocessing of Electrical Schematics Using Picture Graphs. *Computer Vision, Graphics and Image Processing*, 28:271–288, 1984.
4. I. Chai and D. Dori. Orthogonal Zig-Zag: An Efficient Method for Extracting Lines from Engineering Drawings. In C. Arcelli, L.P. Cordella, and G. Sanniti di Baja, editors, *Visual Form*, pages 127–136. Plenum Press, New York and London, 1992.
5. U. Cugini, G. Ferri, P. Mussio, and M. Protti. Pattern-directed Restoration and Vectorization of Digitized Engineering Drawings. *Computers and Graphics*, 8(4):337–350, 1984.
6. D. Dori, Y. Liang, J. Dowell, and I. Chai. Sparse-Pixel Recognition of Primitives in Engineering Drawings. *Machine Vision and Applications*, 6:69–82, 1993.
7. Y. Fukada. A Primary Algorithm for the Understanding of Logic Circuit Diagrams. *Pattern Recognition*, 17(1):125–134, 1984.
8. O. Hori and S. Tanigawa. Raster-to-vector Conversion by Line Fitting Based on Contours and Skeletons. In *Proceedings of 2nd International Conference on Document Analysis and Recognition, Tsukuba (Japan)*, pages 353–358, 1993.
9. Y. Jiafeng, B. CaoQing, T. Agui, and T. Nagao. The Use of Complex Transform for Extracting Circular Arcs and Straight Lines in Engineering Drawings. In *Proceedings of 11th International Conference on Pattern Recognition, Den Haag (Netherlands)*, volume 3, pages 290–293, 1992.
10. T. Kaneko. Line Structure Extraction from Line-Drawing Images. *Pattern Recognition*, 25(9):963–973, 1992.
11. P.C.K. Kwok and T.J. Turner. Raster to Vector Conversion in a Map Interpretation System. In *Proceedings of IAPR Workshop on Machine Vision Applications, Tokyo (Japan)*, pages 165–168, 1990.
12. C.P. Lai and R. Kasturi. Detection of Dashed Lines in Engineering Drawings and Maps. In *Proceedings of First International Conference on Document Analysis, Saint-Malo, France*, volume 2, pages 507–515, 1991.
13. X. Lin, S. Shimotsuji, M. Minoh, and T. Sakai. Efficient Diagram Understanding with Characteristic Pattern Detection. *Computer Vision, Graphics and Image Processing*, 30:84–106, 1985.
14. W. Lu, Y. Ohsawa, and M. Sakauchi. A Database Capture System for Mechanical Drawings Using an Efficient Multi-dimensional Graphical Data Structure. In *Proceedings of 9th International Conference on Pattern Recognition, Rome (Italy)*, pages 266–269, 1988.
15. G. Monagan and M. Röösli. Appropriate Base Representation Using a Run Graph. In *Proceedings of 2nd International Conference on Document Analysis and Recognition, Tsukuba (Japan)*, pages 623–626, 1993.
16. V. Nagasamy and N.A. Langrana. Engineering Drawing Processing and Vectorization System. *Computer Vision, Graphics and Image Processing*, 49(3):379–397, 1990.
17. T. Pavlidis. A Vectorizer and Feature Extractor for Document Recognition.

Computer Vision, Graphics and Image Processing, 35(1):111–127, 1986.
18. C.C. Shih and R. Kasturi. Extraction of Graphical Primitives from Images of Paper Based Line Drawings. *Machine Vision and Applications*, 2:103–113, 1989.
19. K. Tombre. Finding and Coding Graphics in a Composite Document. In *Proceedings of 5th Scandinavian Conference on Image Analysis, Stockholm (Sweden)*, pages 589–596, 1987.
20. K. Tombre and P. Vaxivière. Paper to CAD conversion for mechanical engineering drawings. Rapport interne, Centre de Recherche en Informatique de Nancy, Vandœuvre-lès-Nancy, 1989.
21. P. Vaxivière and K. Tombre. Celesstin: CAD Conversion of Mechanical Drawings. *IEEE COMPUTER Magazine*, 25(7):46–54, July 1992.
22. L.S. Wolfe and J. de Wyze. An Update on Drawing Conversion. *Computer Aided Design Report*, 8(6):1–11, 1988.