Automatic Reading of Technical Drawings : symbol extraction by geometrical and morphological methods

Cécile Dachet
Électricité de France
Direction des Études et Recherches
1, avenue du Général de Gaulle, 92141 Clamart Cedex - France
(+33 1) 47 65 46 50 — e-mail : Cecile.Dachet@der.edf.fr

Abstract

The automatic conversion from pixel images to a format appropriate for a CAD system, requiring a high level of interpretation and composition of its elements, is investigated here for schematic diagrams, which consist mainly of symbols representing components, connecting lines, and text. The major difficulty does not come from the symbol recognition, as they are more or less normalized, but from their location in the diagram. We present a segmentation process that reflects our human approach to the problem : even if one is not used to mechanical or electrical engineering drawings, one is able to distinguish symbols from the rest. It means that some geometrical properties exist that make a symbol a symbol, be it a transformer or a valve. The segmentation is made domain independent by using general tools such as mathematical morphology and simple digital geometry (connexity, convexity, ...). Experimental results on several drawings show that all the symbols can be extracted, within a rather short time compared to the problem size (the drawings size are approximatly 2000 × 1000 pixels), and dealing with the drawings imperfections.

1 Introduction

CAD system utilization is now widespread for designing, storing, consulting and updating of graphical information. But there still is a large amount of paper based documents which require a conversion to be exploited by CAD tools. Manual — or even semi-automatic — conversion is long and fastidious, thus leading to the idea of an automatic conversion.

The difficulty does not come from the symbol recognition, as they are more or less normalized and have simple geometrical shapes (see figure 1), but from their location in the diagram. Moreover, two other problems arise :

- the data size : the drawing sheet size is A1 (84× 60 cm), which makes, with a 200 dpi resolution, more than 10^8 points.
- the quality of the input drawings : they are hand-drawn (with a rule for lines and templates for characters). Defects are numerous and expanded by the scanning process. And, even if human vision system can deal perfectly with topological defaults (broken lines, stuck characters, ...), they are a major hitch to automatic reading.

As a pattern recognition problem, the topic is well-covered in the literature. The general approach was basically syntactic at its early stages in the 70's. It is now acknowledged that one should use mixed techniques, such as expert systems with numerical methods. Many basic methods are involved, and one point in this work is to find out which should be used.

As the analysis of the binary image is based on pixels, it is sometimes valuable (it depends on the following operations) to keep only the most significant ones by extracting the image skeleton. Lee *et al.* [5] presents a thorough review of the topic. There are basically two kinds of methods : finding the pixels at the maximum distance from the borders, or thinning (peeling) the border pixels. When the thickness of the lines is important (or when symbols are made of solid areas), it is possible to extract the contour lines instead of the skeleton (see for instance Pavlidis [9, ch7]).

Vector coding of the enpoints of a line (Smith [14], Pavlidis [10]) is a way to build a graph, whose con-

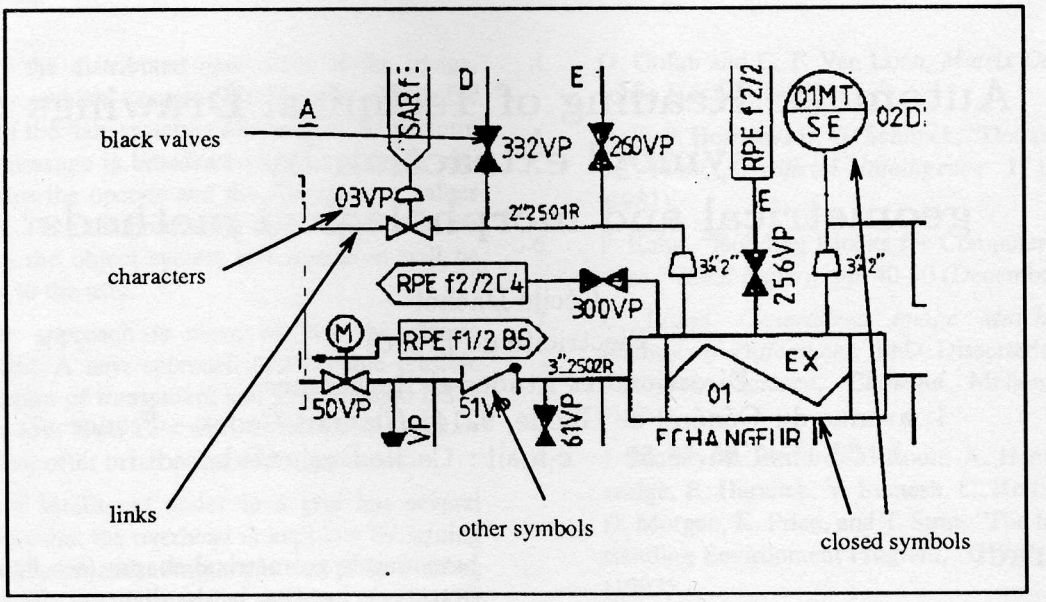

Figure 1: Drawing elements

nections are a key to the identification of the shapes (Miclet [6] gives a general description of this and other syntactical techniques). It is worth noting that, when the components bear specific properties, they can be extracted by other techniques (for instance, Okozaki [7] extracts loop symbols with connected component analysis of the image).

Following another idea, a numerical approach would consider the image (and the symbols in it) to be sets of pixels whose significance lies inside their geometrical properties. Density filters, pattern correlation filters and connected component analysis are generally sufficient to decompose the diagram into pieces. In a second step, selected areas are run into numerical filters (Fourier transform, Hough transform (Pao *et al.* [8]), statistical analysis, histograms, pattern correlation, ...) yielding parameters for any relevant classification technique such as neural networks (Le Cun [4]).

We have already studied and even tried most of these methods. Three guidelines emerge :

- **keep as close as possible to the original raster image.** This ensures that the maximum information is preserved until the recognition process. Image analysis can be context driven with maximum information (thus minimum error) at each decision step. This also allows us to choose any appropriate method at any step, regardless of previous operation. Consequently, classical preprocessings (such as skeletonisation, vectorisation, line regularisation) are useless.
- **use the domain-independent geometrical properties of the drawings.** This reflects our human approach : even if one is not used to mechanical or electrical engineering drawings, one is able to distinguish symbols from the rest. It means that some geometrical properties exist that make a symbol a symbol, be it a transformer or a valve.
- **divide then conquer** : each step reduces the size of the image to be processed, thus allowing more and more refined algorithms for interest areas.

2 Description of the problem

Considering a binary image made up from three classes of components — symbols, connecting lines, and characters — our aim is to realize a segmentation of the image to separate these three classes (see figure 1).

Symbols are :

- closed curves, most of the time convex, sometimes with straight lines and characters inside,
- open curves, but exclusively composed of little straight lines,
- black surfaces (only one kind of symbol : valves).

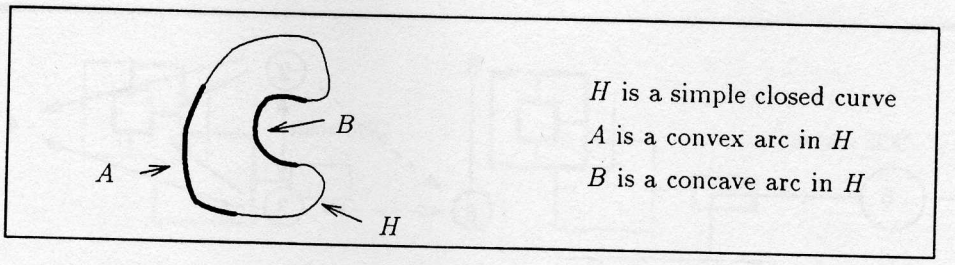

Figure 2: Convex and concave arcs in a curve

Links are most of the time long straight lines, horizontal or vertical.

Characters are little units which in theory do not touch other symbols. Actually, characters often touch other characters, symbols or links.

Our purpose here will be the extraction of the most frequent type of symbol : closed convex curves.

3 Definitions and notations

3.1 Images

A **bidimentional continuous image** is an application from a closed rectangle of $\mathbf{R}^2$ into $\mathbf{R}$, which associates to each point of the plane $\mathbf{R}^2$ the luminous intensity detected by a sensor at this point.

$$I : \; D_I \subset \mathbf{R}^2 \rightarrow \mathbf{R}$$
$$p \rightarrow I(p)$$

If an image only takes values 0 or 1, it is a **binary image**. An image is called a **free bounded image** if all the points of its boundaries have value 0. We shall now consider only bidimensional continuous binary images, free bounded. The set of black points of a binary image is denoted by B, and the set of white points B^c.

3.2 Curves

Let us denote by SCC a simple closed curve of the plane[1]. A SCC is said to be **maximal** if it cannot be included in another SCC of the image. A convex SCC that cannot be included in any other *convex* SCC is said to be **maximal convex**.

[1]We will not go further here in the mathematical description of an image, a curve, convexity ... For information, see [3].

A **simple image** is an image containing only one maximal SCC, its interior, and white points elsewhere.

We denote by $\beta(H)$ the interior of a curve H. Let A be a simple arc included in a closed curve H. A is said to be **convex in** H if and only if :

$$\forall (a,b) \in A^2, [a,b] \subset (H \cup \beta(H))$$

3.3 Surfaces

The set of connected components of a set E is denoted by $\Omega(E)$. The boundary of a set is denoted by $Bd(E)$. Let us consider the set of connected components of the set of white points : $\Omega(B^C)$. The adhesion[2] of one of these connected components is called a **surface**. A set of surface of an image is denoted by $\Sigma(I)$.

$$\Sigma(I) = \{\overline{c}, c \in \Gamma(B^c)\}$$

Moving from a component to its adherence is a means of getting rid of parasitic arcs in the components, and to include the component boundary.

The **initial background** is the surface containing the image boundary. All the other surfaces are said to be **simple surfaces**.

Two surfaces are **neighbouring** if their intersection is not empty.

Giving a set of simple surfaces, if the union of all their points is connected, it is called the **association** of these surfaces. An association is **maximal** if it cannot be included in another association. It means that its only neighbour is the initial background.

A surfaces association is **maximal convex** if it is convex and cannot be included in another *convex* surfaces association.

Let S_i and S_j be two neighbouring surfaces in an image. We say that S_i is in **convex neighbourhood** with S_j if $Bd(S_i) \cap Bd(S_j)$ contains only

[2]a set adhesion consists in all the points having at least one neighbourhood whose intersection with the set is not empty. The notation is $\overline{E}$.

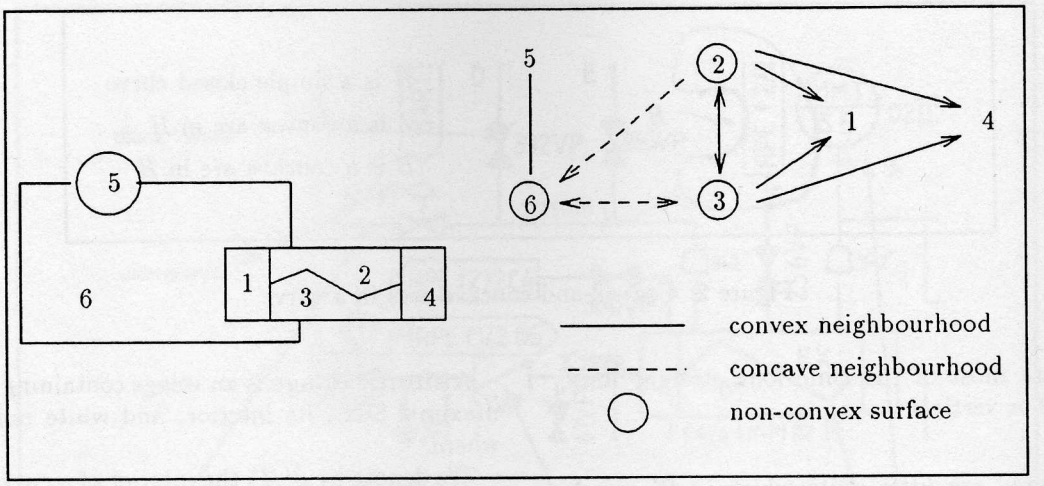

Figure 3: Neighbourhood graph (for non-convex surfaces)

convex arcs in S_j. The notation is *neigh*(S_i/S_j) *is convex*[3]. See figure 3.

Property Let S_1 be a non convex surface, neighbour of S_2, convex or not, so that *neigh*(S_1/S_2) is not convex. If a convex association σ exists so that $S_1 \subset \sigma$, then $S_2 \subset \sigma$.

This property ensures that a non convex surface S_1 can belong to a convex association only if this association contains all the neighbours in concavity of the surface S_1.

3.4 What is a symbol ?

3.4.1 First approach : contour based

> A **symbol** is the union of a maximal convex SCC and of its interior.

3.4.2 Second approach : surface based

> A **symbol** is a maximal convex surfaces association .

It can easily be shown that these two approaches (contour based and surface based) are equivalent. See figure 4 for examples of SCC and symbols.

[3]The relation *neigh*(S_i/S_j) *is convex* is not transitive, nor reflexive, nor symetric.
If S_i is convex, then $\forall S_j$, *neigh*(S_i/S_j) is convex.

4 Symbol Extraction Algorithm

Looking at our definitions, two methods arise :

- enumerating all the SCC, looking for the convex ones, and then for the maximal convex. But in fact, curves of the image are not directly accessible, as the only thing that we know is the value (black or white) of each image point. In an image, there is a lot of SCC and their enumeration is complex.
- trying all the surfaces associations in order to exhibit the maximal convex ones. Surfaces are easily accessible, but trying all the associations is a high combinatory problem for non-trivial images.

The strategy is the following : we first look for all the maximal SCC of the image, that can be rapidly achieved by the procedure described below. Then, for each maximal SCC, there are two possibilities :

- the SCC is convex : it is a symbol, the extraction work is finished.
- the SCC is not convex : it is not a symbol, but maybe there are symbols inside ; a combination of surfaces can be processed as we have now greatly reduced the problem size.

4.1 Maximal SCC search

The aim of this step is twofold : to extract most of the symbols and to reduce the problem size by getting smaller images to work on, without breaking any symbol.

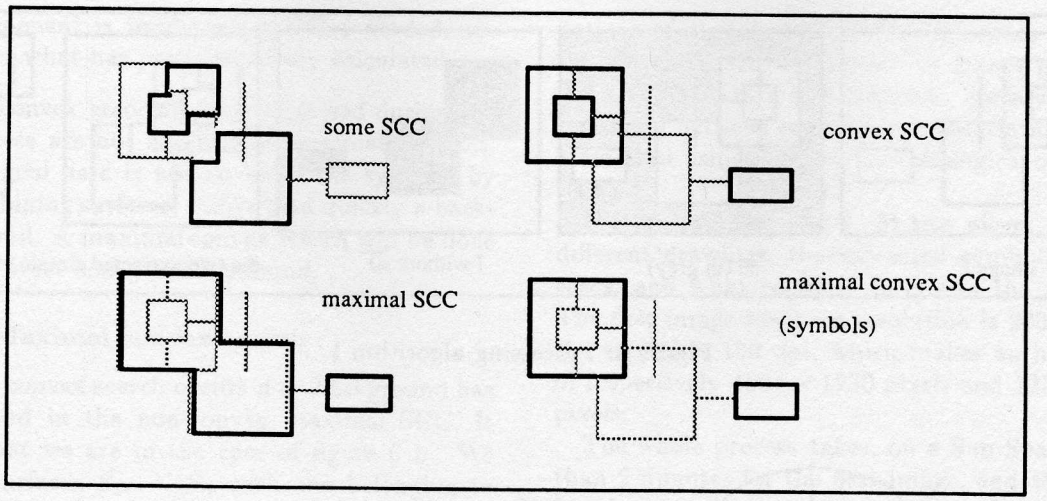

Figure 4: Different kinds of SCC - Symbols

initial state : I is a binary image, free bounded.
goal : the set of all the maximal SCC of I.
algorithm : $A_1(I)$

1. compute s_0 = initial background
2. compute $\overline{s_0}^c$ =
   everything but s_0 and the external curves
3. perform connected component analysis $\Omega(\overline{s_0}^c)$
4. for all $c \in \Omega(\overline{s_0}^c)$, return $(Bd(\overline{c}))$

See figure 5 for an example of processing.

properties no symbol is broken. In fact, if a symbol is lost during the process, it means that a symbol made of two parts belonging to two different maximal SCC exists. As a symbol is made of neighbouring surfaces, these two SCC must be neighbouring. And that is impossible, coming from the very definition of a SCC. Thus we are ensured that no symbols will be broken by this decomposition into simple images.

proof sketch

1. every g in $A_1(I)$ is a maximal SCC of I :
   as s_0 is connex and $Bd(s_0) = Bd(D_I)$, we can show that any enclosing curve of g would yield an enclosing connected component of c (where $g = Bd(\overline{c})$).
2. every maximal SCC belongs to $A_1(I)$:
   we show that $\beta(g)$ belongs to $\Omega(\overline{s_0}^c)$ because $\beta(g)$ is connex and any enclosing connected component s' of $\beta(g)$ would define an enclosing SCC $g' = Bd(\overline{s'})$ of g.

4.2 Splitting of the non-convex maximal SCC

4.2.1 Splitting algorithm

Given a simple image with non-convex SCC (such as obtained by algorithm 1), there are two possibilities :

- all the surfaces belong to a symbol. The problem is to constitute the different symbols (see figure 6 b),
- there exist some specific surfaces (by definition non convex) which belong to no symbol ; it means that some symbols are not neighbouring with the initial background : symbols are interlocking (see figure 6 a). These specific surfaces bear the same properties as the initial background, and we can move back to algorithm 1 by replacing the initial background by the union of the initial background and specific surfaces.

initial state : a simple image I.
goal : all maximal convex SCC of I (symbols).
algorithm : $A_2(I)$

1. search for a surface that is a *background*
   (algorithm 3) $s = A_3(I)$
2. **if**
   s = success (one is found)

then
   back to algorithm 1 with CC_0 =
   initial background + surface found

else
   maximal convex search $A_4(I)$(see 4.2.3)

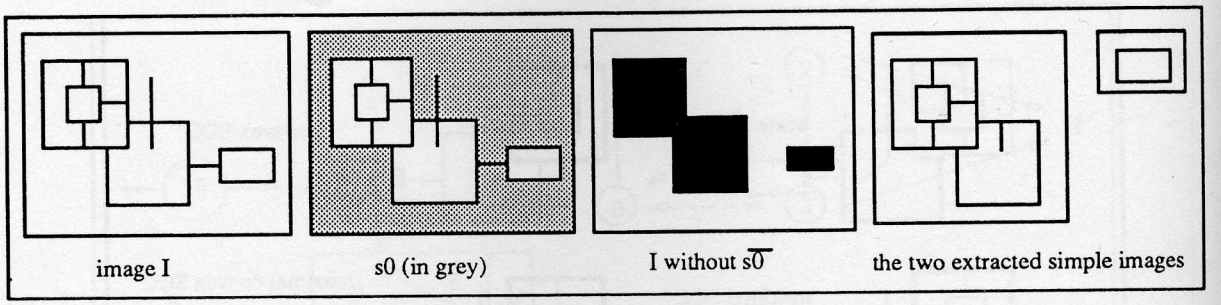

Figure 5: Processing algorithm 1

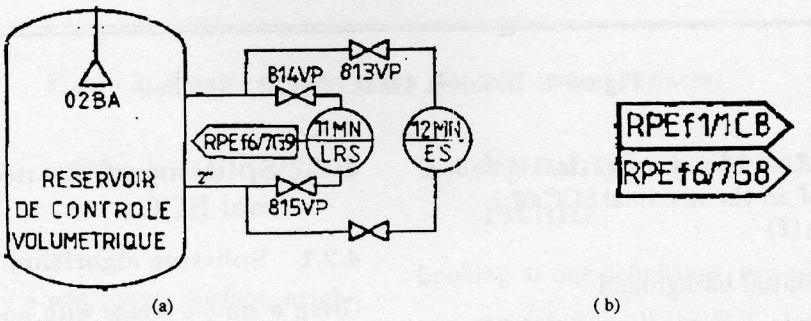

Figure 6: (a) interlocking symbols (b) connected symbols

4.2.2 Background search

The goal of this stage is to examine non-convex surface in order to determine if they can be a *background* or not. A surface is a *background* if it is not part of a symbol, that is to say, if it is impossible to form a convex association containing it. In order to reduce the amount of surfaces combinations to test, some preliminary results have been established in 3.3.

Using the property of 3.3, the algorithm to find a background is the following.

initial state : a simple image I
goal : a background or a failure
algorithm : $A_3(I)$

1. perform component labelling
   on the white pixels $\Omega(B^c)$
2. classify the components (except the initial background) by decreasing area
3. **for** each $s_i \in \Omega(B^c)$
   if
   s_i was not memorized as belonging to a convex association
   and s_i is not convex
   then
   $S \leftarrow \overline{s_i}$+ all the neighbours in concavity of $\overline{s_i}$
   if
   it is possible to form a convex surface by adding to S surfaces from the convexity neighbourhood of $\overline{s_i}$
   then
   memorize this association
   go to the next s_i
   else
   return s_i as a background
   endif
   endif
4. return no background found

properties

1. the faster the background is found, the smaller the number of components to be scanned is, as we stop as soon as a background is found. Moreover, in practice, the background is often bigger than any other surface. By examining the components in decreasing area order, we speed up the search.

2. the neighbourhood graph is not built entirely at the beginning. The neighbourhood of each

component is implemented when needed, reusing what has previously been calculated.

3. the convex associations memorized during the process are not necessarily maximal. In fact, the goal here is not to build the symbols by combining surfaces, but to find quickly a background. A maximal convex search will be done only if no background is found.

4.2.3 Maximal convex search

Maximal convex search occurs if no background has been found in the non-convex maximal SCC. It means that we are in the case of figure 6 b. We have n surfaces $S_1, \ldots, S_n$ each one belonging to symbols. The aim is to discover these symbols, that is to say the maximal convex associations.

The preceding step sometimes gives us some memorized convex association that can be reused.

Starting from a surface, the algorithm proceeds with neighbourhood association until reaching a stable state where all different symbols are found. Maximal convex association are built step by step by adding new surfaces which preserve the convexity. In theory, such a construction does not yield to a unique solution, but in practice we have never met any problem. We will not describe here the whole algorithm, as there is nothing special to be pointed out.

4.3 Some remarks on the method

One can wonder why we have not, from the beginning, worked on surfaces, by a component labelling on the white pixels. This is the method employed in [7] for electronic circuit diagram reading.

First of all, in our case, detecting all the CC is not sufficient to have all the symbols : some are built by association of many, even non convex, surfaces. And we cannot afford a search for all surface associations on the image, because of its computational complexity. Reducing considerably the size and number of images where this is done is one of the interests of our process.

Moreover, letters inside symbols are often stuck to the border, thus leading to some problems in the evaluation of surfaces ; our filling method, coming from the exterior of the symbol, is not disturbed by these little surfaces.

5 Implementation and results

The search methods described in the euclidian case are adapted to the geometry of a digital image. The notions of digital connected components and digital convexity are well-known in image processing (see [1] and [11]). Adherence, surfaces, neighbourhood between surfaces, and association are implemented using geodesic morphological operations ([13]).

We present here (fig 7, 8) two pieces from two different drawings, the extracted symbols in both cases, and what remains on one of the drawings. The first image scanning resolution is 200 dpi, and the second is 150 dpi, which makes an image size of respectively 1964×1230 pixels and 1256×2006 pixels.

The whole process takes, on a Sun Sparc 2, less than 2 minutes for the first image, and 45 seconds for the second one. Although the size of the first image is smaller, the processing time is longer because it is really much more complex : many lines are forming non-convex SCC that have to be resegmented. We present in figure 9 some of the simple images being extracted.

Note the robustness of the process against parasitic lines (see for example sensor[4] 095RN on figure 7), and touching characters.

6 Conclusion

We have presented here a segmentation process to extract symbols from mechanical drawings (i.e. pipe and instrumentation drawings). The process is particulary interesting because :

- at each step, the problem size is reduced, allowing more and more acute analysis on complex areas,
- except their convexity, no additional information on the symbols to extract is needed. This made the process domain-independent,
- the fact that the process comes from the "exterior" of symbols makes it very robust to parasitical lines, connected characters, etc.
- the original image is kept intact : the information is not modified as for example a vectorization could have done it. This is a major advantage for the recognition step of the symbol.

Two negative points have to be admitted : (1) the process is sensitive to broken lines ; a symbol with a hole on one side will be missed. (2) the line thickness must be approximately known and big gaps will lead to mistakes.

[4] A sensor is represented by a circle.

As we have already said, extraction is the major issue for symbols : once they have been extracted, the recognition process is quite simple ; shape indices are sufficient enough to identify symbols. Most of the label strings that identify the symbol are within it and are thus extracted during the same process. The reading of the label strings has been experimented (using neural-nets) and already gives good results. Some symbols (valves, for example) are made of two independent convex loop that have to be associated by a distance criterium. Their label string is outside and must also be associated by a search in the neighbourhood of the symbol.

What remains on the image is then some special symbols (composed by little non-closed line), links, and some characters. In order to classify these three classes, the presence and size of straight lines can be used. Furthermore, the fact that most of the symbols have already been detected gives some hints for the following steps : a line that leaves a symbol is a link.

Once all the symbols are identified, the links known, and the label string read, moving to a CAD format can be performed.

References

[1] J.M Chassery. Géométrie discrète en analyse d'images. *Hermès*, Paris, 1991.

[2] U. Cugini *et al.* Pattern directed restoration and vectorization of digitized engineering drawings. *Computer and Graphics*, v. 8, n. 4, pp 337–350, 1984.

[3] J. Dieudonné. Eléments d'analyse. Gauthier-Villars, Paris, vol 1, 1972.

[4] Y. Le Cun *et al.* Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1, 541–551 (1989).

[5] S.W. Lee, L. Lam and C.Y. Suen. Performance evaluation of skeletonization algorithms for document image processing. *in Proc. First Int. Conf. on Document Analysis and Recognition*, Saint-Malo, France, pp 260–271, 1991.

[6] L. Miclet. Méthodes structurelles pour la reconnaissance de formes. Éditions Eyrolles, Paris, 1984.

[7] A. Okozaki *et al.* An automatic circuit diagram reader with loop-structure-based symbol recognition. *IEEE-PAMI*, v. 10, n. 3, 1988.

[8] D. Pao, H.F. Li and R. Jayakumar. Graphic features extraction for automatic conversion of engineering line drawings, *in Proc. First Int. Conf. on Document Analysis and Recognition*, Saint-Malo, France, pp 533–541, 1991.

[9] T. Pavlidis. Algorithms for graphics and image processing. *Computer science press*, Bell Laboratories, New Jersey, 1982.

[10] T. Pavlidis. A vectorizer and feature extractor for document recognition. *Computer vision, graphics and image processing.*, 35, 111–127, 1986

[11] A.Rosenfeld, A.C.Kak. Digital Picture Processing *Academic Press*, New York, 1976.

[12] J. Serra. Image Analysis and Mathematical Morphology. *Academic Press*, London, 1982.

[13] M. Schmitt, J. Mattioli. Mathematical Morphology *Technical Report, Thomson CSF, LCR*, Orsay, 1992.

[14] R.W. Smith. Computer processing of line images : a survey. *Patt. Recogn.*, 20, pp. 7–15, 1987.

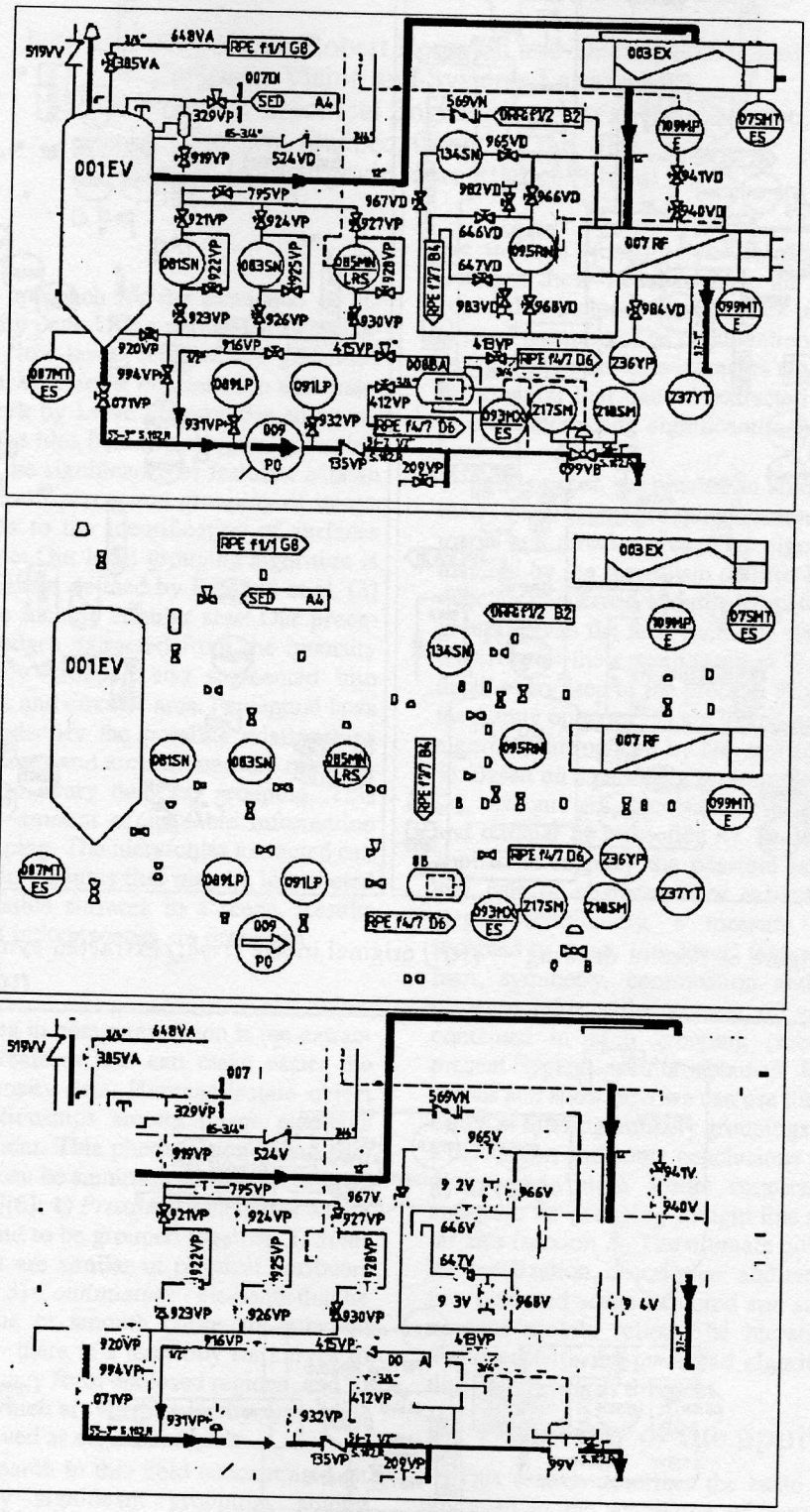

Figure 7: First drawing — *(top)* original image *(middle)* extracted symbols *(bottom)* remain image

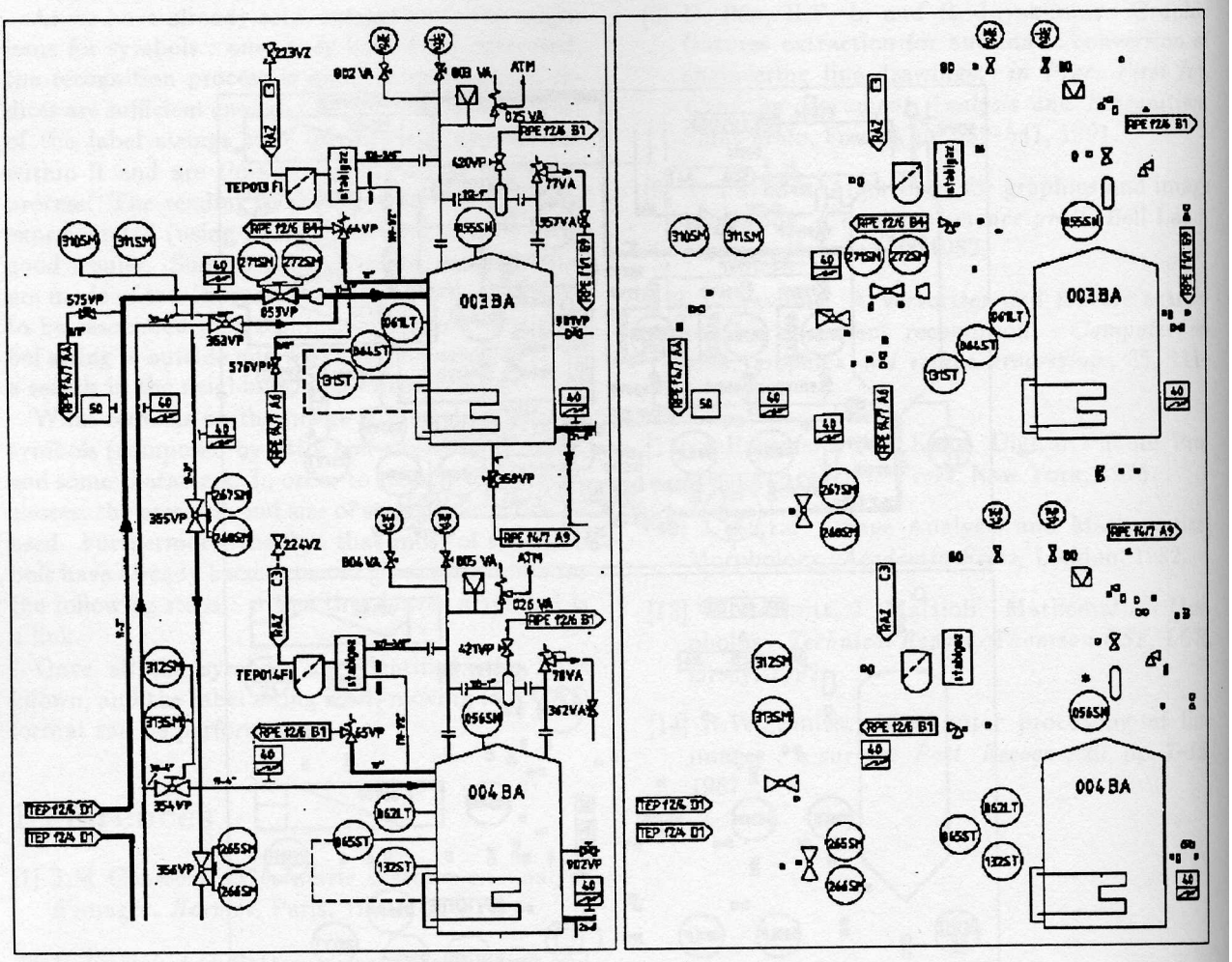

Figure 8: Second drawing — *(left)* original image *(right)* extracted symbols

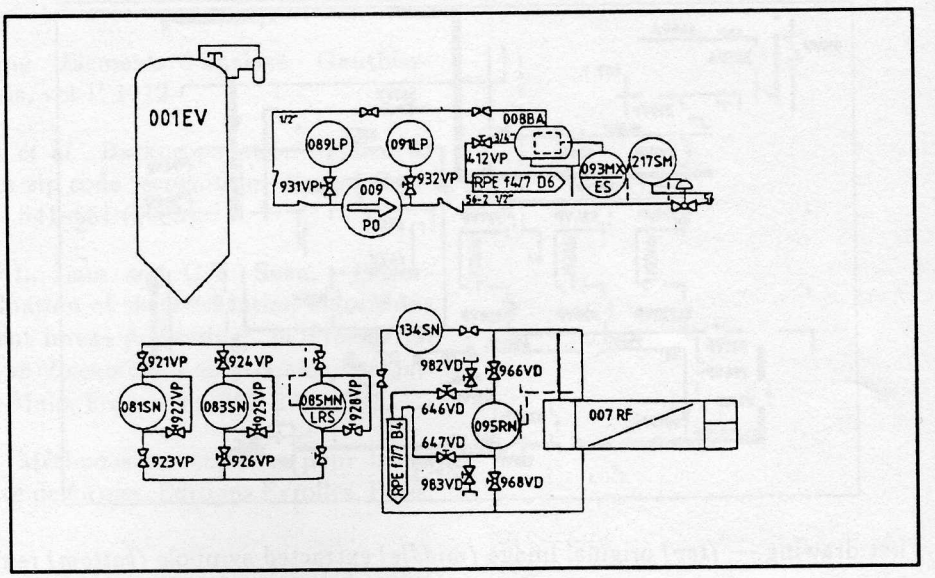

Figure 9: Examples of simple extracted images