Published in *Advances in Handwriting and Drawing : a multidisciplinary approach,* C. Faure, P. Keuss, G. Lorette, A. Winter (eds), pp. 21-38, Europia, Paris, 1994.

Extracting text lines in handwritten documents by perceptual grouping

Laurence Likforman-Sulem, Claudie Faure
Télécom Paris - CNRS - URA 820 (FR)

ABSTRACT: We here present a new method for segmenting handwritten pages of documents into text lines. The method is an iterative process which uses some clues from the physiology of vision and Gestalt laws of perceptual organisation. These clues include direction detection and perceptual grouping based on proximity and direction continuity. The segmenting process takes into account specific problems of handwritten documents such as different line directions appearing within a page, lines overlapping or being interwoven.

KEYWORDS: handwritten documents - segmentation - text lines

1. Introduction

The automatic analysis of handwritten documents is of great importance for desktop publishing, archiving and storage. The challenge is to provide a high level representation with which to manipulate the document obtained from a scanned image. Other applications concern manuscript authentication, the paleographic study of texts and forensic document analysis (PLAMONDON and LORETTE, 89; LIKFORMAN-SULEM *et al.*, 91). The primary goal is not character recognition but the extraction of characteristics such as the makeup of the text, the writing style or the identity of the writer. The third class of applications concerns mail sorting and automatic reading of forms and bank notes wherein specific items like postal code or amounts, located at predictable places, must be recognized.

In this context, we shall study handwritten documents, i.e. whole pages of manuscripts including both rough drafts and letters. Recognizing a page of manuscript first involves interpreting its layout structure, which means extracting blocks, lines of text, and words. The description of a document's content in terms of signature, address blocks, date or sections, refers to its logical structure. In this work we will focus on the extraction of the lines of text relative to the document's

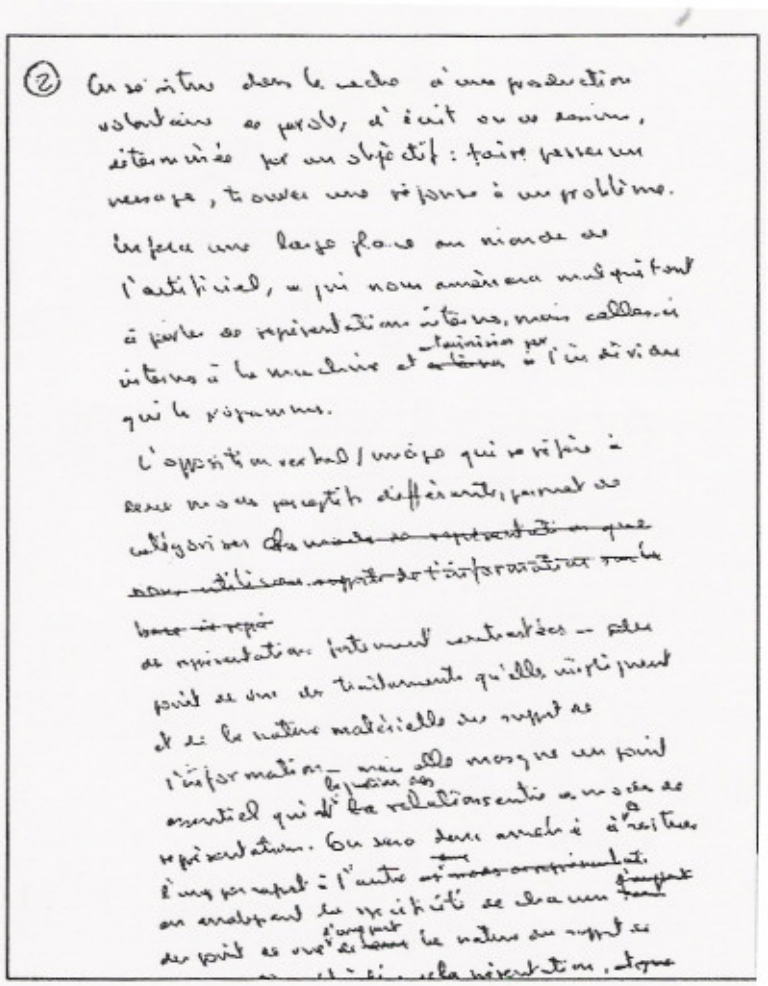


Figure 1. *Handwritten page of document*

physical structure.

This structural analysis is problematic in handwritten documents due to the fact that the spacing between words and lines is irregular, two lines may overlap or be interwoven, handwriting is fluctuating rather than straight, words may be inserted between lines, and different line directions may appear within a page. Connected components of the image generally do not correspond to single words as words may be broken into several parts or on the contrary, two words may be linked. Moreover words or lines may be crossed out. Some of these characteristics can be found in figure 1.

The segmentation of handwritten texts has been little studied so far because recognition methods deal with isolated words or characters. Applications devoted to handwritten texts concern postal code extraction or the analysis of regular or simplified texts (COHEN *et al.*, 91; PAQUET *et al.*, 89; DOWNTON and LEEDHAM, 90; SHAPIRO *et al.*, 93).

The new method of segmentation presented here aims at extracting lines on a whole page of manuscript. For handwriting, spread across a whole page of text, no strong model can be used for predicting the location of information. We assume here that extracting text lines is a low level visual process that does not need any recognition of the text (this is not true for segmenting into words). At a distance (or half closing the eyes), lines are visible. The alignment detection method proposed here is an iterative process which uses some clues from the physiology of vision and Gestalt laws of perceptual organisation (WERTHEIMER, 23). These clues include direction detection and perceptual grouping based on proximity and direction continuity.

First we shall give some definitions about alignments and neighborhood, as well as an overview of the method's steps. Afterwards we will describe each one of these steps and provide results in comparison to more classical methods which have been developed for printed documents.

2. General Description and Definitions

The method is an iterative process composed of several steps:

1- Connected components labeling
2- Selection of directional anchor points
3- Construction of alignments
4- Evaluation of the quality of the alignments
5- Conflict resolution

Step 1 is a classical image processing routine which extracts connected black point units. A connected component may be a word, a part of a word, a character, a

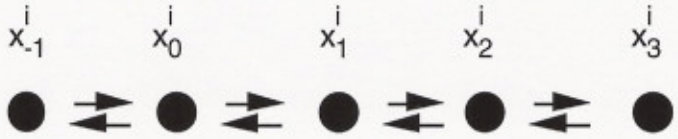
part of a character or a group of words if they are linked by an erasure or if they touch each other. Step 2 is the selection of connected components which have a reliable direction. This is a local direction which can either be opposed or not to the global direction of the text line. Each anchor point constitutes the seed of an alignment. The third step of the process consists in linking other neighboring components to anchor points by perceptive criteria. Then the quality of the alignments is evaluated. The fifth step of the process consists in solving alignment conflicts by applying rules relative to their spatial configuration and quality.

At the end of the fifth step, anchor points have been extended to stronger alignments. The sequence composed of steps 3, 4 and 5 is iterated, the distance threshold being incremented at each iteration. The process is iterated until complete alignments are found.

We define Ω_c as the set of connected components found in the image. An alignment A_i is a spatially ordered set of connected components, linked together by 2 possible relations:

$\rightarrow$ is the negative neighbor of

$\leftarrow$ is the positive neighbor of



Alignment i is noted $A_i = \{ x^i_k \in \Omega_c, k=-m \text{ to } +n \}$

We define Ω_A as the set of alignments in the page: $\Omega_A = \{ Ai , Ai \text{ has at least one element}\}$

x^i_0 is the anchor point, seed of alignment i (see section 3). The other connected components have been linked during the grouping process (see section 4).

For each connected component, we define E_{xp} as the set of alignments to which a connected component belongs, that is: $E_{xp} = \{ Ai, xp \in Ai\}$

At the beginning of the process $E_{xp} = \emptyset \ \forall xp \in \Omega_c$. During the segmentation process, each connected component is given one or several labels depending on the number of alignments it belongs to. If card $E_{xp} > 1$, the component is *ambiguous*, i.e. it belongs to more than one alignment. On the contrary, if card $E_{xp} = 0$, the component xp is still isolated.

The goal of the segmentation process is to obtain card $E_{xp} = 1, \forall xp \in \Omega_c$. Iterations cease when alignments become stable.

3. Orientation Detection

Hubel and Wiesel have shown that in the human visual system, in the primary visual cortex, in area 17, there are simple cells which are sensitive to line segments and their orientation. The receptive fields of these cells are rectangular (figure 2-a) having an excitatory and an inhibitory area (BUSER and IMBERT, 86). The response of the receptive fields depends on the intensity of the excitation. We use the properties of receptive fields to analogously define masks that will detect the local direction of a connected component which may be a word, a part of a word, or a character. The masks are sensitive to an excitatory signal, in the present case handwriting (i.e. black pixels). Although we consider square masks rather than rectangular ones, the excitatory zone remains rectangular.

The space is sampled into 4 directions (0°, 45°, 90°, 135°) and a mask is set up for each direction. Each mask is a square circumscribing the component (figure 2). These masks are adapted to the dimensions of the components. . The 4 masks are applied to each connected component. The mask response is the black point density of the component within the mask:

$$\text{response} = \frac{\text{number of black pixels in the excitatory zone}}{\text{number of black pixels of the component}}$$

If a component gives a high response (superior to a chosen threshold, here 90 %) to a mask, it is considered as an anchor point whose direction is that of the mask (figure 3).

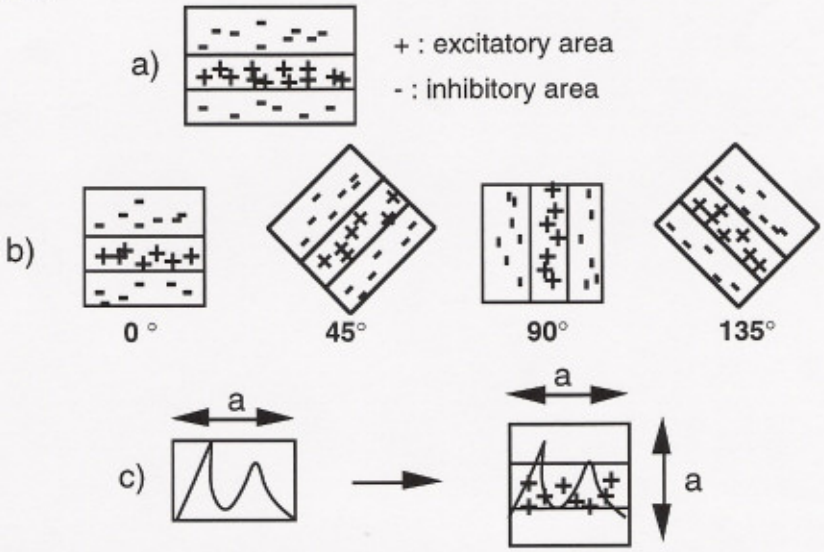


Figure 2. *a) Receptive field of a simple cell in the primary visual cortex*

b) 4 masks are defined for each direction

c) application of a horizontal mask to one component

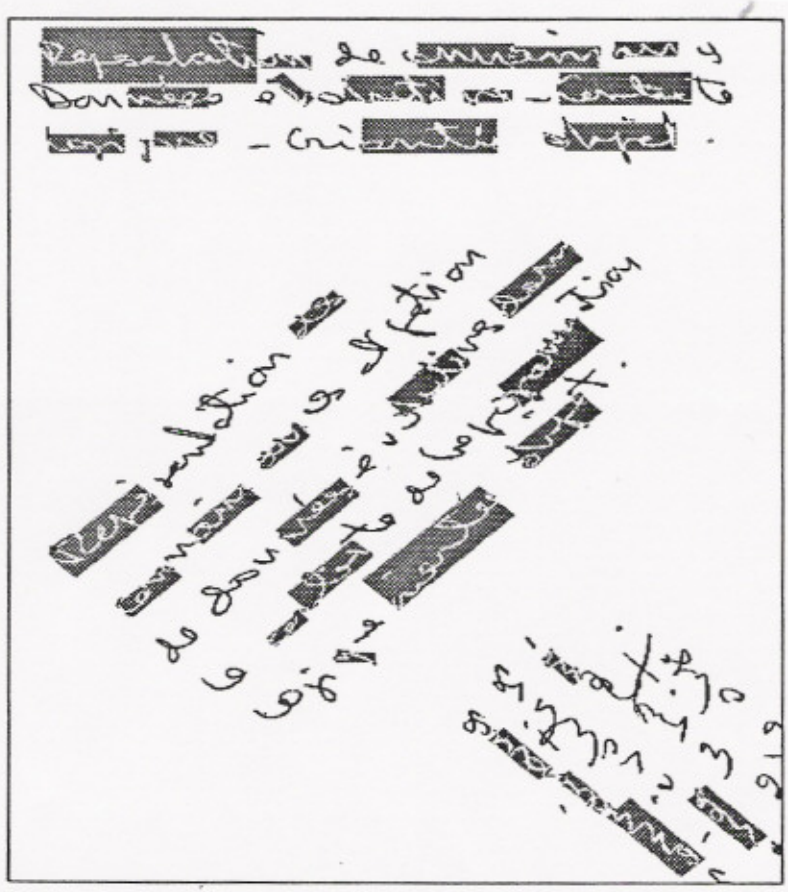


Figure 3. *Handwritten document (detail). Selection of anchor points (highlighted), each one constitutes the seed of an alignment.*

4. Grouping

From the features given by the cells, the visual system aggregates these features into higher order ones. Gestalt theory has given a number of principles of perceptual organisation that enable us to perceive certain aggregates of elements. We have retained two grouping principles for alignment construction: proximity and direction continuity (FAURE and LIKFORMAN-SULEM, 93). Elements close together are grouped together. The second point is that perceptual organisation tends to preserve smooth continuity rather than yielding abrupt changes. The aggregates we seek to obtain in this work are the alignments of connected components.

Each detected anchor point is now the seed of an alignment. Thus if K anchor points have been selected, K alignments still reduced to their anchor point are built:

$$Ai = \{ x_0^i \} \quad i = 1 \text{ to } K.$$

Other neighboring components have to be linked to anchor points (step 2) by perceptive criteria, i.e. components which are within a threshold distance D (proximity) and in the direction of the anchor point (direction continuity). The search for a neighboring component is performed by an image processing routine which looks for a neighbor in the image, whether in the *positive* or in the *negative* neighborhood as indicated in figure 4. Positive and negative neighborhood are generic terms specifying image regions at each side of a connected component. Size and localisation of these regions depend on the height of the component itself, the proximity threshold D and the direction of the alignment to which the component belongs. The relationship between component x_k^i and its positive neighbor x_{k+1}^i is noted by the two symetric relations :

$x_{k+1}^i = \text{neighbor}_1^+ (x_k^i)$ and $x_k^i = \text{neighbor}_1^- (x_{k+1}^i)$

Number 1 indicates that the *first* nearest neighbor was searched. Signs + and - indicate the relative positions between the two components (in the positive or in the negative neighborhood).

Once a component is linked, another is sought from it. Components in the positive neighborhood of anchor points are searched first. When the next component is beyond the proximity distance or when the edge of the page is reached, components in the negative neighborhood are sought.

At the first iteration, the distance threshold is very small. This threshold is extended at each iteration by a small fixed value Δ. As of the second iteration, neighboring components are not sought from anchor points but from the first and the last components of alignments. This enables us to avoid fixing *a priori* a distance threshold which would have to fit the data.

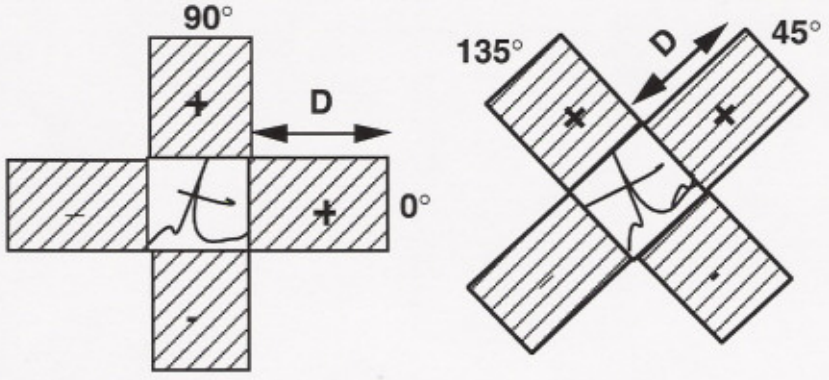


Figure 4. *Positive and negative neighborhood of a connected component for directions 0°, 90°, 45°, 135°*

The segmentation task can be seen as a component labeling process. Each component is given zero, one or several labels depending on the number of alignments it belongs to. The label is the identification of the alignment. During the construction of an alignment, the label of the alignment is attached to the components linked as follows:

if a positive neighbor $x^i_{k+1} \in \Omega_c$ is found for component x^i_k belonging to alignment Ai with,

$Ai = (x^i_{-l}, ..., x^i_{-1}, x^i_0, x^i_1, ..., x^i_k)$, then Ai becomes:

$$Ai = (x^i_{-l}, ..., x^i_{-1}, x^i_0, x^i_1, ..., x^i_k, x^i_{k+1}) \text{ and } E\,x^i_{k+1} = Ex^i_{k+1} \cup \{ Ai \}$$

If no positive neighbor is found, a negative neighbor is sought for x^i_{-m}. If it exists, then Ai becomes: $Ai = (x^i_{-m-1}, x^i_{-m}, ..., x^i_{-1}, x^i_0, x^i_1, ..., x^i_k)$ and $E\,x^i_{-m-1} = E\,x^i_{-m-1} \cup \{ Ai \}$

5. Alignment Quality Evaluation

The quality of each alignment is evaluated (step 3) in order to keep relevant alignments and eliminate others at step 4. After each iteration, linked components can re-enforce or weaken alignments obtained from the previous iteration. We consider 3 values for computing alignment quality. These values quantify the quality and the visibility of the alignments according to the spatial organisation of elements. The first value tests the global aspect of the alignment, while the latter two test the alignment's local aspect. They are:

-a) NC is the number of components of the alignment. The longer the alignment, the stronger (the more visible) it is.

-b) MD is the number of anchor points included in the alignment which have the direction of the alignment. When the global direction of the alignment corresponds to local directions, this re-enforces the alignment.

-c) DD is the number of anchor points included in the alignment which do not have the direction of the alignment.

The best alignments are obtained for high NC and MD values and low DD values. From these 3 values, a single numerical probabilistic value P is computed for each alignment:

$$P = P_1 * P_2$$

$$\text{with } P_1 = \frac{1}{1 + \dfrac{k_1}{NC-2 + \dfrac{MD}{NC}}} \quad \text{if } NC > 1, \quad P_1 = 0 \text{ otherwise}$$

and $P_2 = \frac{1}{1 + \frac{D\,D}{k_2}}$

k_1 and k_2 are integer parameters, varying from 1 to 10, which can be tuned depending on handwriting and text characteristics. Low k_2 values fit texts containing numerous symbols such as parenthesis, or numbers. Such symbols generate anchor points whose directions are opposed to that of the text line. From these anchor points, alignments in opposite directions will quickly grow. The quality of such alignments must be lowered in order to eliminate them during the conflict resolution step.

The quality of an alignment depends on its length which is indirectly measured by the number of components (NC). When handwriting is fragmented, as in script, short alignments contain several connected components which yield high P1 values. The quality of these alignments can be decreased, by increasing k_1.

From value P, alignments are classified into three classes: strong, weak and intermediate. The quality factor is used further in the conflict resolution phase to solve conflicts between competing alignments.

6. Conflict Resolution

When two alignments meet on one component X, card $E_X > 1$ and there is a conflict. We separate cases (figure 5) according to whether the conflictual alignments have the same direction (a-b-c) or not (d). We solve these conflicts by applying a set of 4 logical and quantitative rules, and search routines in the image (step 5). The conditions of the rules consider elements as alignment configuration, their quality and direction continuity. The actions of the rules consist in merging, splitting, extending or deleting alignments.

6.1 *Alignments of same direction*

Figure 5.a) exemplifies the *Y-fork* spatial configuration. We suppose below that the alignment direction is horizontal. The first nearest neighbor of components c1 and c3 is X :

$X = neighbor_1^+(c1)$

$X = neighbor_1^-(c3)$

We have also $c2 = neighbor_1^+(X)$

In order to configure the alignments, search processes are activated to find second nearest neighbors for c1 and c3. They are respectively denoted y and z :

$y = neighbor_2^+(c1)$

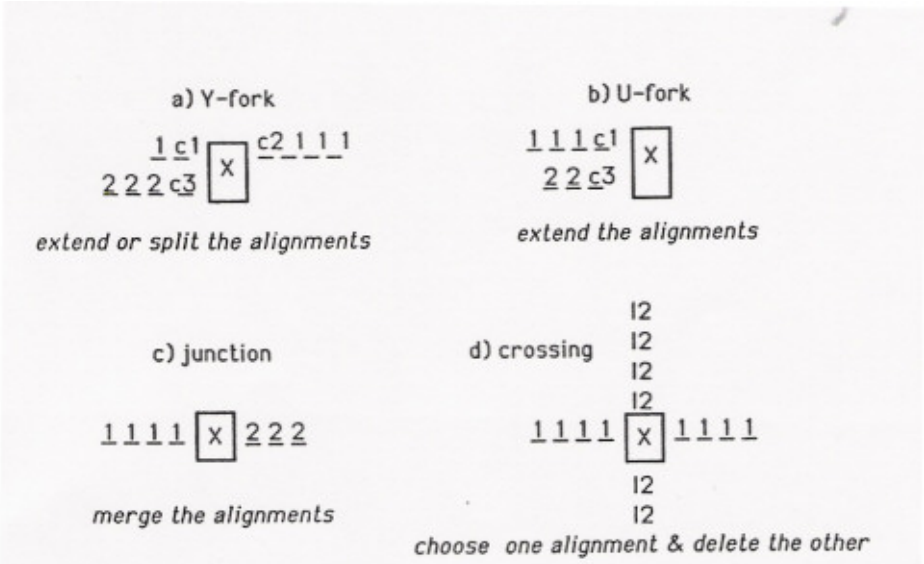


Figure 5. *Conflicts occur when alignments meet. Different configurations may occur when alignment 1 meets alignment 2 on component X.*

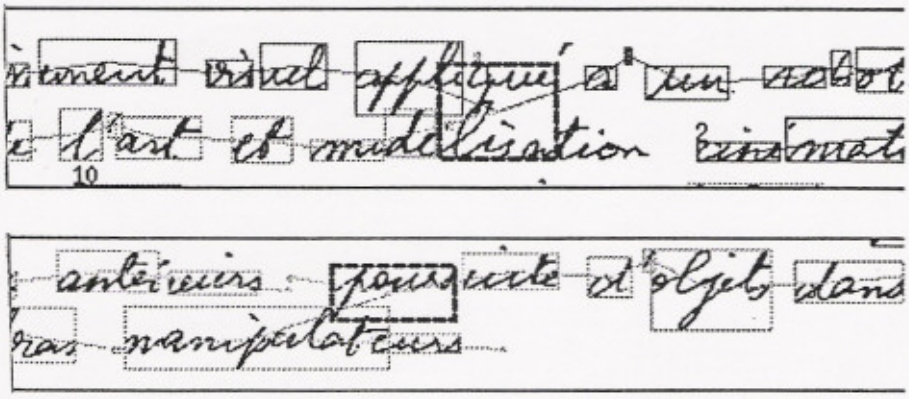


Figure 6. *Two cases of Y-fork configurations : components at the alignment intersections are in bold rectangles. Lines 1 and 2 are overlapping, lines 3 and 4 are interwoven.*

$z = neighbor_2^+(c3)$

We only modify alignments 1 and 2 in reliable cases, by the following rules. first one is:

IF $y=neighbor^+_2(c1)= c2 \in A_1$,
$z=neighbor^+_2(c3) \neq \emptyset$
THEN $X \in A_1$, $X \in A_2$ and $z \in A_2$

The action of the rule consists in extending alignment 2 with z and kee component X in both alignments.

The second one is:

IF $z=voisin^+_2(c3) = c2 \in A_1$
THEN $X \in A_2$, $c2 \in A_2$ and labels of all the components of A_1 on the right are changed in label 2.

Alignment 1 is split and alignment 2 chosen for X.

In all other cases, alignment 1 is chosen for component X.

These rules enable us to build alignments even if lines overlap (X results the overlapping of two components belonging to different writing lines), shapes are interwoven (figure 6). At present, the process detects such proble components X. Such information will be of great importance during a recogr phase. Further treatment will consist either in choosing one of the alignmen component X or in splitting component X into an upper part and a lower part.

When the *U-fork* configuration occurs (figure 5-b), the same search proces in the above configuration are activated in order to extend one or both alignn But in this case, three other neighbors (denoted respectively y, z and k) are sea for components c1, c3 and for component X. Neighbor k is searched in an ext neighborhood in order to check whether X is on the limit of the writing are have :

$X = neighbor_1^+(c1)$
$X = neighbor_1^+(c3)$
$y = neighbor_2^+(c1)$
$z = neighbor_2^+(c3)$
$k = neighbor_1^+(X)$

If X is on the limit of the writing area, X remains in both alignmen applying one of the two following rules :

IF $y = neighbor_2^+(c1) = \emptyset$
$z = neighbor_2^+(c3) = \emptyset$
$k = neighbor_1^+(X) = \emptyset$
THEN $X \in A_1$, $X \in A_2$
or
IF $y = neighbor_2^+(c1) \neq \emptyset$
$z = neighbor_2^+(c3) \neq \emptyset$
$y \neq z$
$k = neighbor_1^+(X) = \emptyset$
THEN $X \in A_1$, $X \in A_2$, $y \in A_1$, $z \in A_2$

Alignments A_1 and A_2 are extended on y and z. If the two neighbors are equal, one alignment is arbitrarily chosen to be extended.

If one only of the alignments may be extended, one of the following rules can be applied :

IF $y = neighbor_2^+(c1) \neq \emptyset$
$z = neighbor_2^+(c3) = \emptyset$
$k = neighbor_1^+(X) \neq \emptyset$
THEN $X \in A_1$, $X \in A_2$, $y \in A_1$
or
IF $y = neighbor_2^+(c1) \neq \emptyset$
$z = neighbor_2^+(c3) = \emptyset$
$k = neighbor_1^+(X) = \emptyset$
THEN $X \in A_2$ and $y \in A_1$

The case $y=\emptyset$ and $z\neq\emptyset$ is processed symetrically.
By applying the first rule, we get a Y-fork configuration which will be solved at the next iteration. If the second rule is applied, A_1 is extended on y and X only belongs to A_2.

The *junction* configuration (figure 5-c) is the meeting of two parts of the same alignment. The two parts are merged together.

6.2 *Alignments of different direction*

When two alignments of different direction cross each other (figure 5.d), the absolute quality of each one and their relative strength are tested from the quality factors computed at step 4. If one of the alignments is of good quality and significantly stronger than the other, it is chosen and the other is deleted.

7. Iterating the process

Once step 5 is achieved, we get alignments which are generally still incomplete (corresponding to parts of text lines only). The process is iterated, the distance threshold being incremented, to link other components to the alignment. At the first iteration, the distance threshold is small in order to link components belonging to a same word or to words very close together.This distance may be a few pixels or a longer distance depending on the resolution of the text images. At each iteration new components have to be reached by incrementing the distance threshold by a fixed value Δ. The tuning of Δ also depends on the resolution. If the fixed value is too low, one has to wait for more iterations before constructing complete alignments. If the fixed value is too high, a lot of components will be quickly aggregated in wrong and long alignments. These alignments will hardly be eliminated at step 5.

The number of iterations necessary to get complete alignments depends on the spacing between components belonging to the same text line. We get complete alignments typically between 3 and 6 iterations. When alignments are strong enough and do not cross any other alignment, they get a final labeling which will not be changed by the following iterations.

8. Results

It is difficult to provide a quantitative evaluation of the segmentation method. Hence, the method is qualitatively compared with other classical ones, and results are given on several kinds of images.

8.1 *Comparison with other methods*

Our results are compared with other methods such as projections, Hough transform and smearing (WONG *et al.*, 82; FLETCHER and KASTURI, 88; PAQUET *et al.*, 89). These methods are the most popular ones for line detection.

Projection based methods assume only one text direction and are sensitive to line fluctuation, to lines that are close together and a fortiori to lines which are overlapping. After having determined the number of black pixels along the projection axis, lines are determined by searching for an alternance of peaks and valleys of this projection profile. This model is accurate when all lines share the same direction, when they are not too short and when lines are not interwoven: they are sufficiently spaced so that the position of all downstrokes from one line is clearly above the position of the upperstrokes from the following line. In figure 7 b, lines are not correctly found because they do not fulfill this last requirement.

The run length smoothing algorithm, or smearing algorithm, consists in linking together neighboring black units which are separated in the horizontal direction by less than a threshold distance. This threshold has to be appropriate, typically the

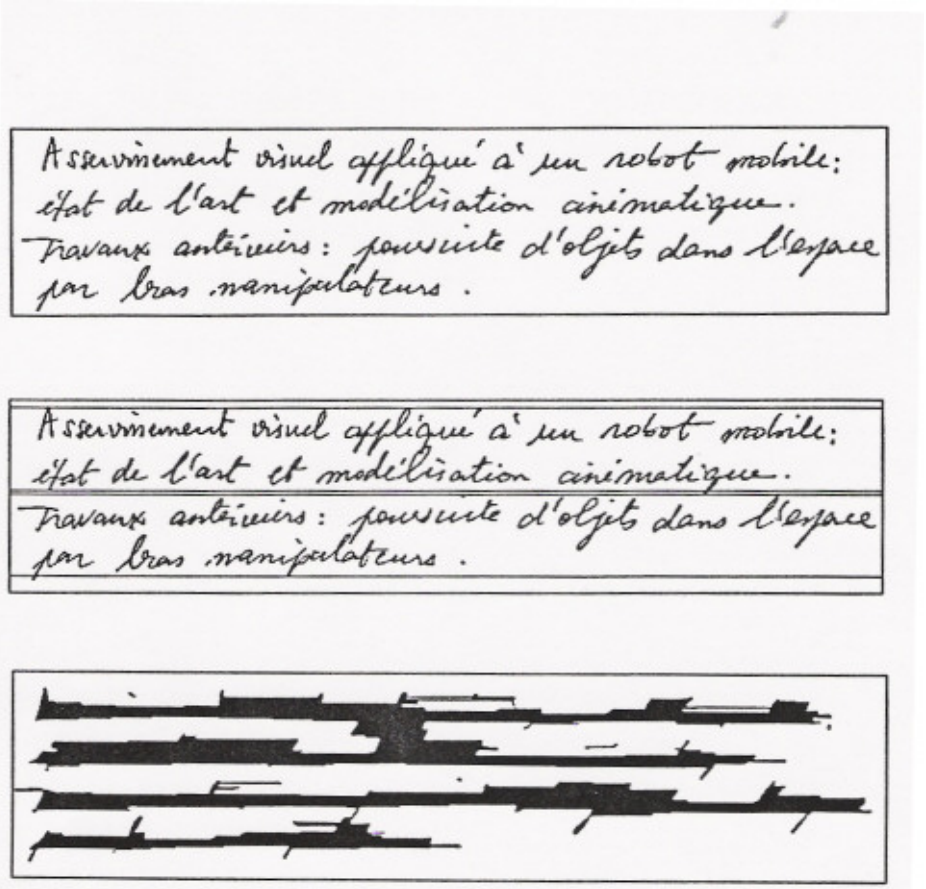


Figure 7. *a) detail of handwritten document*

b) projection method. Lines found by the method are surrounded by two black lines.

c) smearing method. The 2 first lines and the two last ones are connected after smearing.

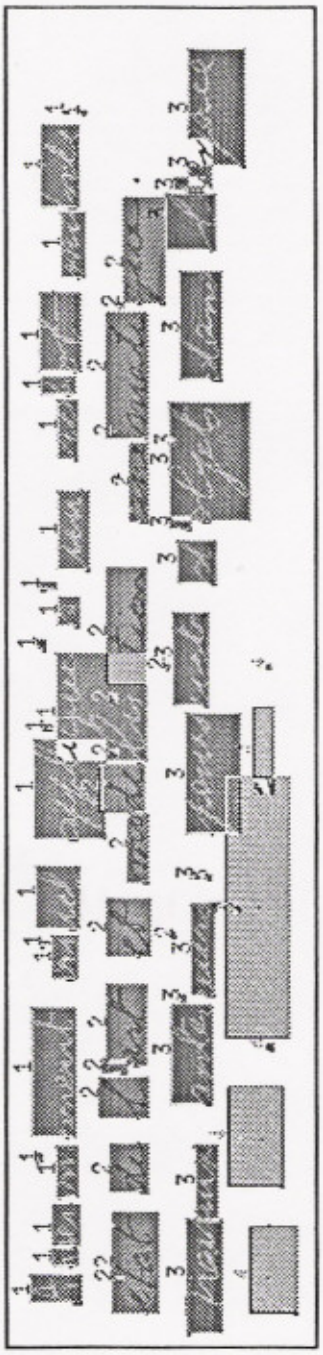


Figure 8. *Hough transform method. applied to figure 7 a) The alignment's identification number is inscribed above each connected component.*

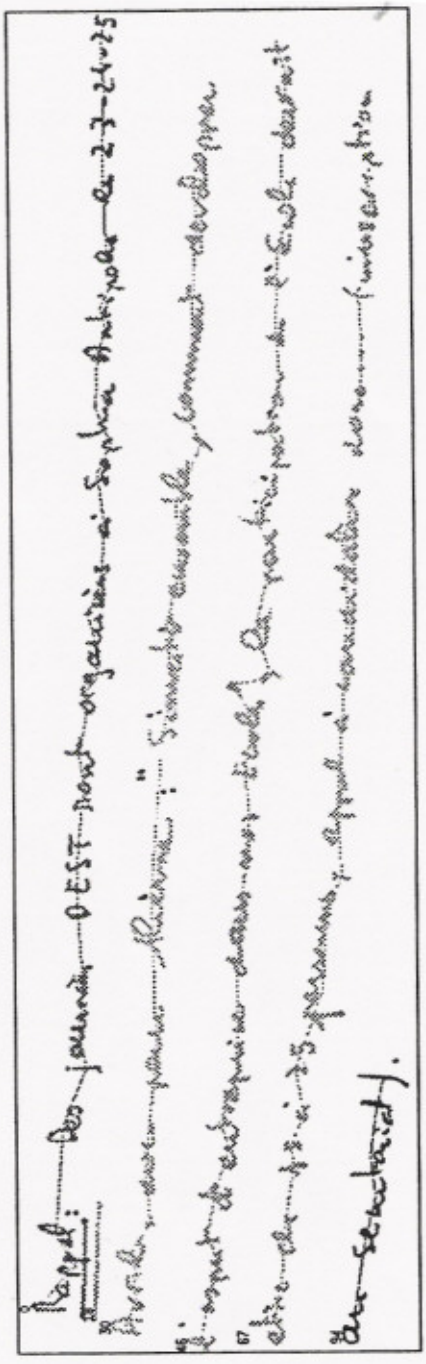


Figure 9. *Alignments obtained on a fluctuating handwriting*

inter-word distance, to link words together to form a line. Moreover, the difficulty is that a fixed fusion criterion may not fit all parts of a handwritten document. Secondly lines must be sufficiently spaced as in projection methods to avoid linking together words located on two different lines. In figure 7 c, lines 1 and 2, and lines 3 and 4 are connected after smearing.

The Hough transform is less sensitive to characteristics such as orientation and lines close together, but is sensitive to handwriting fluctuations. The transform detects points (for instance the center of gravity of connected components) situated on a same straight line, in any direction. The stronger alignments will be found first, i.e. lines that include a greater number of points. Irrelevant alignments may be detected such as a set of components belonging to different writing lines. For instance, in a page where the text is written horizontally, Hough may find long oblique alignments crossing the page. A solution would be to force the detection of alignments in an a priori chosen direction as in figure 8. This figure shows the alignments found for figure 7 a). In this figure, an overlapping component contains parts of words belonging to line 1 and line 2. This component is included in line 1 and definitively excluded from line 2.

8.2 *Sensitivity to noise and resolution*

Small sized components are eliminated as noise during the connected components labeling. Eliminated components should have a size inferior to the typical size of an accent or a comma. Moreover, components whose size is about that of an accent are eliminated as anchor points in order to start the line detection process with more reliable anchor points.

Image resolution is chosen as a compromise between image storage and the preservation of thin strokes as up and downstrokes. In the range of tested documents which are handwritten pages and also handwritten postal addresses, size thresholds are chosen according to image resolution. A resolution of about 150 ppi is a good compromise for the tested documents. If the resolution is too low, handwriting will be over-fragmented and less anchor points will be found.

9. Conclusions

Our method deals with handwritten pages including lines going in several directions. Line detection methods such as projections and smearing need an a priori given direction before processing the whole document. Results obtained with Hough transform are greatly improved if alignments are also searched in an a priori given direction. Our method is able to detect lines of writing in several directions by processing stepwise the whole document from local to global. Most conflicts which arise when a component belongs to two different lines are detected and solved during the process. If not, components belonging to two lines are labeled as ambiguous and will need further processing.

Satisfactory results are obtained when enough anchor points in the right direction

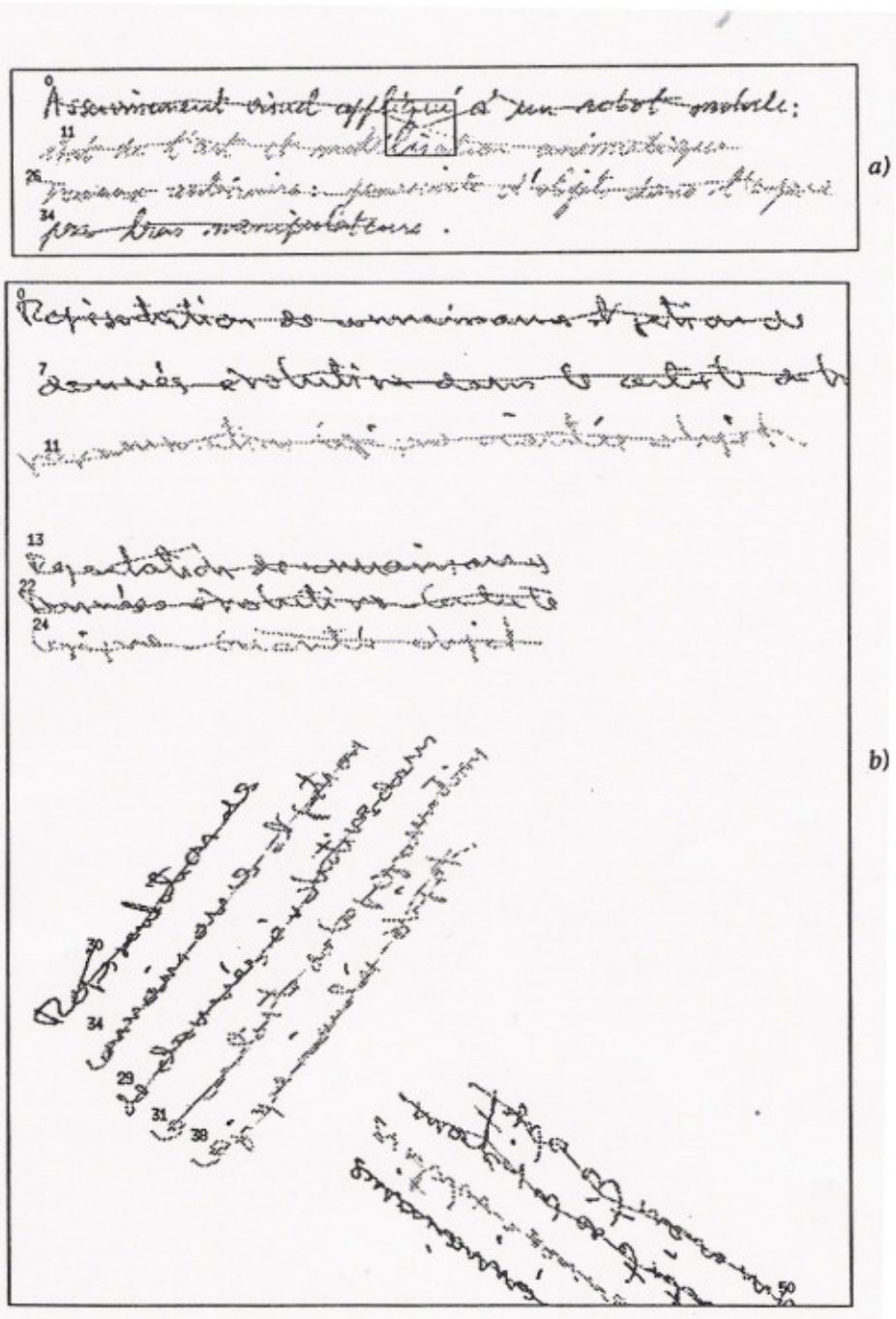


Figure 10. *Alignments obtained after a few iterations. Alignments are in different grey levels. All the components of an alignment share the same label written at the beginning of the alignment.*

are found within each text line. On the other hand, if no anchor point is found, the components belonging to the text line are left unlabeled. Dots and accents may also be left unlabeled. In figure 10, alignments are in different grey levels and the label of the alignment is indicated at the beginning of each one. In figure 10-a, lines are correctly found after solving two cases of Y-fork configurations (shown in figure 6). The common component should be split into two parts. The incremental increase of neighborhood size allows one to follow fluctuating lines. Figure 9 is the result of the segmentation on a fluctuating handwriting.

Grouping errors appear when the line direction has a value between the sampling directions.We can refine the process by adding other directions and by using contextual information such as dominant direction within blocks, writing height and interline distance. This would enable the detection of possible errors and the possibility of guiding further steps of the segmentation process.

References

BUSER P., IMBERT M., 1987, Vision, Herman, pp.404-436.

COHEN E., HULL J., SRIHARI S., 1991, Understanding handwritten text in a structured environment : determining zip codes from addresses, *Int. Jour. of Pattern Recog. and AI,* Vol. 5, No 1 & 2, June, World Scientific, pp.221-264.

DOWNTON A.C., LEEDHAM C.,1990, Preprocessing and presorting of envelope images for automatic sorting using OCR, *Pattern Recognition*, Vol. 23, No 3-4, pp. 347-362.

FAURE C., LIKFORMAN-SULEM L.,1993, Traitement automatique de l'écrit : structuration perceptive et catégorisation, *Textuel No 25*, Ecrire, Voir, Conter, pp.37-54.

FLETCHER L.A., KASTURI R., 1988, Text string segmentation from mixed text/graphics images, *IEEE PAMI*, Vol 10, No 3, pp. 910-918.

LIKFORMAN-SULEM L., MAITRE H., SIRAT C., 1991, An expert vision system for analysis of Hebrew characters and authentication of manuscripts, *Pattern Recognition,* Vol 24, No 2, pp.121-137.

PAQUET TH., MULLOT R., TRUPIN R., ROMEO K., LECOURTIER Y., 1989, Un algorithme rapide de détection des mots d'un texte manuscrit, Congrès AFCET-RFIA, Paris, pp. 1501-1510.

PLAMONDON R., LORETTE G.,1989, Automatic signature authentication and writer identification: the state of the art, *Pattern Recognition,* Vol 22, No 2, pp. 107-131.

SHAPIRO V., GLUHCHEV G., SGUREV V., 1993, Handwritten document image segmentation and analysis, and methods, *Pattern Recognition Letters* , No 14, pp 71-78.

WERTHEIMER.M., 1923, Untersuchungen zur lehre von der Gestalt II, *Psychologishe Forshung,* Vol 4, pp. 310-350.

WONG K., CASEY R., WAHL F.,1982, Document analysis system, *IBM Journal of research and development,* 26, No 6.