# Erasure Extraction in On-Line Captured Paper Forms

Alain Wiart, Thierry Paquet, Laurent Heutte

# Erasure Extraction in On-Line Captured Paper Forms.

*Alain WIART* alain_wiart@yahoo.fr
*Thierry PAQUET* thierry.paquet@univ-rouen.fr
*Laurent HEUTTE* laurent.heutte@univ-rouen.fr

PSI-LITIS, Université de Rouen,
F-76800 Saint Etienne du Rouvray, France

## Abstract

*In this paper, we describe a preprocessing system which locates erasures in on-line captured handwritten documents. Our approach is conceived so as to be placed upstream of the handwritten recognition engine. This system classifies each couple of connected strokes using a low level feature set and a multi layer perceptron classifier. One part of this study gives an efficient definition of erasure, which results in splitting the two original classes of the problem into nineteen more accurate sub-classes. The tunable tolerance level of the system provides a good flexibility to operate in accordance with various recognition engines. We evaluate our system on a real document database and present encouraging performance results.*

**Keywords**: Erasure detection, on-line recognition, pen based applications.

## 1. Introduction

According to the digitization mode of handwriting, temporal information is available, or not. Unlike off-line scanned paper documents, on-line devices directly capture pencil movements so that a chronological description of the strokes can be restored. This temporal representation gives opportunities to avoid some difficulties of off-line recognition : it makes the recognition of overlapping characters easier, holds information on handwriting dynamics, and remain clear from some inherent distortions of an off-line digitization (variations of the stroke thickness, handwritings damaged by background of page...) [3]. On-line recognition approaches have thus better performance than off-line approaches in the general case [9]. However, the strictly temporal ordering of the on-line signal causes some difficulties, notably for the detection of diacritical and punctuation marks, or erasures. As the chronological order of these components is not always coherent with the reading order, their treatment constitutes an important work in on-line recognition approaches. In this field, many methods have been proposed so as to cope with diacritical and punctuation marks, but the proposed methods concerning erasures processing are quite rare. Indeed, the on-line acquisition mode being mainly implemented within the framework of interactive applications (PDA, tablet PC), the interface often provides a mean to correct by cancellation some strokes. However, some new on-line acquisition devices, like those based on the *Anoto*® technology [11], require to reconsider the problem of erasure detection. This technology, based on the association of a pen with electronic vision and paper printed with an invisible dot pattern, offers the advantage of reconciling on-line digitization with paper documents. The pen is able to know in real time its position on the sheet, by video snapshots of the pattern portion bordering its ballpoint. It is thus able to restore the on-line digitization of handwriting in the form of a temporal sequence of strokes, where all the information relating to the dynamics of writing is stored. It is an interesting alternative to the use of devices such as tablet PC, which could be inappropriate in certain circumstances, due to its size or its weakness. With this kind of technology, the recognition phase is generally delayed from the capture phase. This prevents from any user-supervised recognition. In most of the cases, interactivity is technically impossible, and would not be very compatible with applications that require a fast handwritten capture.

For this kind of on-line capture in particular, it is preferable not to send the strokes corresponding to erasures to the recognition engine without cautions. The strictly chronological sequence of strokes in the signal may involve a misinterpretation of the erasures by the recognition engine. Indeed, when the writer applies corrections while writing, correcting strokes succeed to corrected strokes in many configurations. Without a dedicated preprocessing step, the correspondence between correcting and corrected strokes is difficult to establish. Those strokes will be sent separately to the recognition engine even if its act in a same writing event. It is thus desirable to make the detection of erasures as autonomous as possible, upstream of the handwritten recognition.

We present such a system in this article. Our system is totally independent of the recognition engine. It scans every page to locate the possible existing erasures, before the recognition step. This type of preprocessing

is particularly helpful for some categories of documents. In fact, the context of our study falls under the mass-processing of medical forms. Collected thanks to the *Anoto*® technology, thousands of forms are centralized in a storage server, and then analyzed to produce statistical studies. Automatic reading is then largely beneficial. In their professional environment, medical attendants have to fill out these forms quickly, so that substance is favored at the expense of style. Consequently, various corrections and erasures are regularly observed in these forms, which constitute a notable source of errors if they are not detected beforehand.

In the first section of this paper, we give an overview of the works proposed for corrections processing, and more generally works dealing with the processing of delayed strokes processing. The technical and methodological aspects of our approach are described in the section two, in which we try to propose a way to define the ambiguous concept of erasure. The results obtained with this first approach are analyzed and discussed in the third section of the paper.

## 2. The erasure detection problem.

The on-line recognition approaches exploit a temporal sequence of strokes, transmitted according to an order which is not always in accordance with the reading order. The case of diacritical marks gives a good illustration of this matter: some strokes, such as the crossing of a "t" or the dot of an "i", may appear randomly in time. As the relative positions of strokes are not explicit in an on-line capture signal, the detection of such ambiguous delayed components is not immediate. Consequently, various methods have been proposed so as to take account of diacritical marks: assimilation of diacritical marks to characters by adding all the induced alternative spellings to the lexicon [5] [7], removal of diacritical marks after setting a flag in the feature vector [4], combination of on-line and off-line information through the addition of features extracted from bitmaps [8], or by combining on-line and off-line recognition engines [1], and at last, searching for of the best fitting of delayed strokes during the recognition process, using a forward search algorithm [6].

Works about delayed strokes due to erasures are quite uncommon. Most on-line recognition systems are user supervised applications, so that the recognition results can be checked in real time by the user. Thus, automatic processing of human corrections is quite useless in interactive applications: user can perform immediate corrections by cancelling strokes, or may directly modify the content recognized if he needs to correct older documents.

In the field of interactive applications, ergonomics is more likely to be improved, notably for human corrections handling. Some solutions are more suitable for pen based applications than a "press-button" procedure. In this direction, some systems use a limited set of predefined pencil gestures as correction commands (RESIFCar) [2].

Other approaches propose to improve flexibility by restricting user constraints, such as the error handling intelligent user interface dedicated to NPEN++ [10]. This interface is able to handle simultaneously every kind of delayed strokes, which could be diacritical marks, punctuation marks, or human corrections. In that method, each stroke is segmented on its horizontal extrema. This segmentation allows characterising delayed strokes, that are responsible of a singular regression on the horizontal axis. The repairing strokes, which result from human corrections, are then extracted from the delayed strokes collection using a set of heuristics. When the system identifies a potentially repairing action of the writer, it tries to determine if the aim is to delete or to complete another stroke. Throughout this process, control is kept by the user, so that if a correction is not interpreted correctly by the system, then the user will reiterate his repairing action. The system is able to interpret such repeated repairing patterns as misclassification notifications, and will memorize it. The most interesting aspect of this erasure processing system is its semi-autonomous capability. The machine adapts itself to human correction strategies, not the opposite.

For the applications based on *Anoto*® technology, after the page has been written, digitized strokes are restored by the pen according to their writing time ordering. If no action is made as a preliminary, strokes will be transmitted according to this order to the recognition engine. In the case of an erasure, assuming a word is written and corrected immediately, the recognition engine will probably come to a conclusion about the identification of the word, but will leave away the correcting stroke. This word will appear in the contents resulting from the recognition, whereas the user wished to eliminate it. In addition, if the correct layout is written a long time after the concerned word, another problem arises. For the case of the diacritical marks, the recognition engine would deal with limited variations in the temporal distribution of strokes. However, the task becomes more complicated if strokes in one word do not follow one another immediately in time. It is then necessary to identify within the page several strokes corresponding to the same event, by considering only their geographical positions.

A solution to these problems is to locate all the erasures of the document during a preprocessing stage, so as to anticipate errors that could be induced in the handwriting recognition process. The aim of our study is thus to conceive a recognition system dedicated to erasure identification, which will scan documents before the handwriting recognition process.

In our context, the handwriting recognition phase is delayed from the time of writing and must be carried out on huge amounts of documents, centralized on a server. User interactivity is then forbidden, so our system must be fully autonomous. In addition, since no constraint

was imposed on erasure patterns, the system must ideally adapt itself to every human correction strategies. We thus conceived an erasure recognition system, getting its knowledge from the analysis of real documents.

Our knowledge dataset is extracted from a few hundreds of documents. An analysis of this sample quickly highlights that the erasure concept is ambiguous: some corrections are patent erasures, but others result from localised shape improvements made by the writer. A precise categorization becomes essential to provide an accurate definition of erasure. This is why we made the choice to subdivide the two classes "erasures" and "standard writing" into subclasses (fig. 4). Twenty one subclasses are defined, rather than simply labelling the data as an "erasure" or a "standard writing". Each subclass is supposed to gather similar cases of erasures or similar cases of standard writing, thus allowing more compact regions in the feature space. It is also possible to act accurately on the system behaviour by gathering in various ways these subclasses, for example by moving one of the subclasses from the erasure class towards the standard writing class.

The first distinguishing of erasure features is their correction modes. Some corrections aim at cancelling the strokes they cover, and are called “suppressive erasures” (fig. 1a and 1b). Other ones, known as “overwritings” come to complete words (fig. 1c and 1d). In the first case, all the strokes localised in the erasure zone must be suppressed. In the second case, the correcting strokes are constitutive of the word morphology. It will then require more processing, such as a selective suppression of strokes, a temporal reorganization, or a fusion of strokes by an off-line recognition approach.

We proceed then to other categorizations. In the case of the suppressive erasures, we consider criteria such as the correcting stroke density (dense erasure: fig. 1a, or crossing out erasure: fig. 1b), its extent of action (complete erasure: fig. 2a, partial erasure: fig. 2b or sub-erasure: fig. 2c). For the case of overwritings, we consider the intention of the correcting stroke: if it comes to replace (suppressive overwriting: fig. 1d) or to complete one or more characters (completing overwriting: fig. 1c), or if it acts as an overload, i.e. if the writer repeated an overwriting to make it prominent (fig. 3).

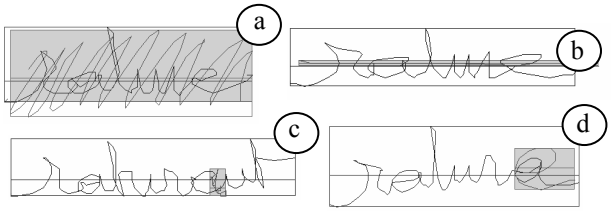



**Figure 1**. Suppressive erasures, and overwritings.

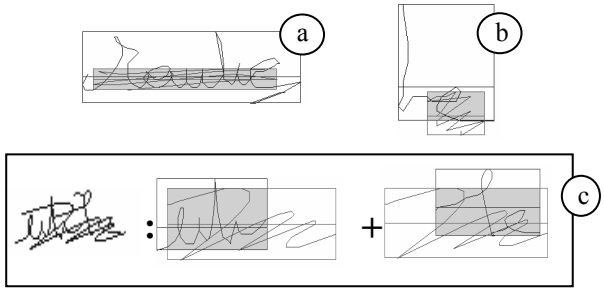



**Figure 2**. Various erasure’s extent of action.

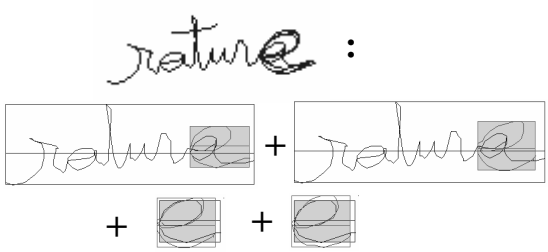


**Figure 3**. overwriting and overloads.

We also proceed to a subdivision of the "standard writing" class, which will distinguish cases of connected strokes dues to diacritical marks, crosses, signatures and sketch, overruns between lines. Eventually, nineteen subclasses are preserved after fusion of under-represented categories (“partial crossing out” and “erasures overruns”: see table 1).

A direct classification of the couples of strokes among these 19 subclasses would be too ambitious. We hope at least to be able to distinguish erasures cases from standard writing cases, and possibly, suppressive erasures cases from overwriting cases.

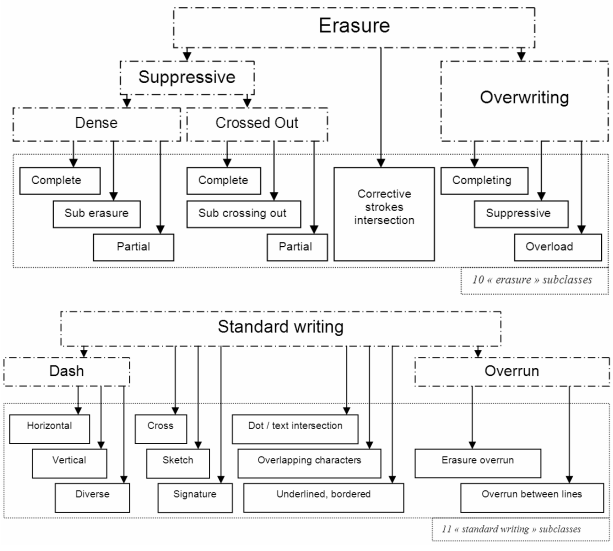



**Figure 4**. Subclasses hierarchy.

**Table 1.** Number of examples in each subclasses.

| Erasures | | Standard writing | |
|---|---|---|---|
| Complete dense | 10 | Horizontal dash | 1669 |
| Complete crossing out | 2 | Vertical dash | 190 |
| Dense sub-erasure | 141 | Various dash | 40 |
| Sub-crossing out | 34 | Cross | 1017 |
| Partial dense erasure | 7 | Signature | 84 |
| Partial crossing out | 0 | Sketch | 127 |
| Corrective strokes intersection | 3 | Character | 1336 |
| Suppressive overwriting | 142 | Erasure overrun | 5 |
| Completing overwriting | 37 | Overrun between lines | 20 |
| Overload | 263 | Underlined, bordered | 35 |
| | | Dot / text intersection | 106 |
| **Total** | **609** | **Total** | **4529** |
| **Total : 5138** | | | |

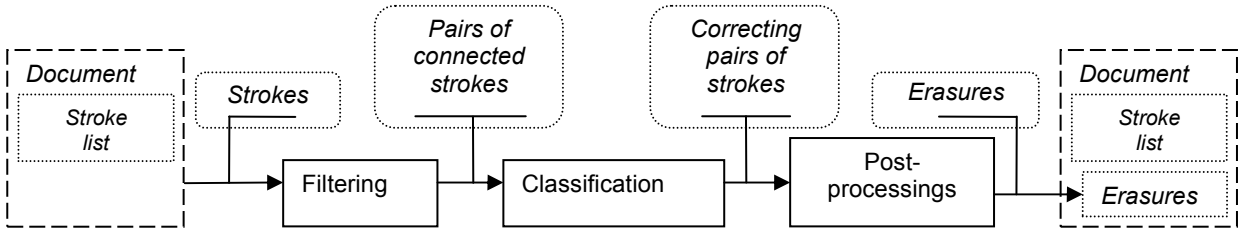



**Figure 5.** Erasure detection scheme.

## 3. Methodology

Our method is based on the assumption that an erasure must spring from the overlapping of several strokes. During a first phase, strokes unconnected with other ones are discarded from the data. Only the strokes presenting one or more connections with their neighbors are taken into account and patched together by pairs. The system must classify these pairs of strokes. We now describe the various stages of the recognition process (fig. 5).

### 3.1. Filtering

A stroke is defined as being the continuous portion of handwritten layout collected from the instant when the pen hits paper to the instant when it is released from paper. Supposing that at least two intersecting strokes are needed to obtain an erasure, it is useless to search for erasures among the strokes completely isolated from their geographical neighbours. For each document, the strokes collection is thus filtered to decrease the number of candidates to the erasure, by removing isolated strokes (fig. 6).

Filtering is carried out according to two criteria. The first, very fast, consists in searching for overlappings between stroke bounding boxes. The intersections between strokes are then searched in this selection. During this operation, the strokes that have at least one intersection between them are coupled together by pairs (fig. 7). The filtering stage gives a new representation of the document in the form of a sequence of stroke pairs. Within each pair, the second (latest) stroke is the potentially repairing one.

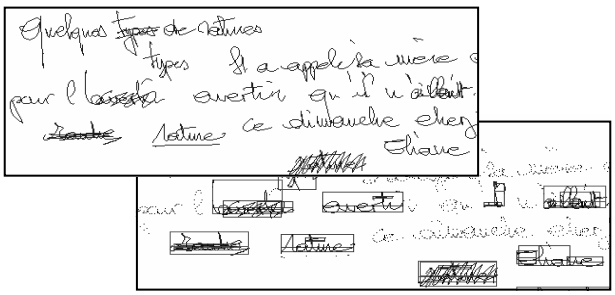


**Figure 6**. Isolated strokes filtering.

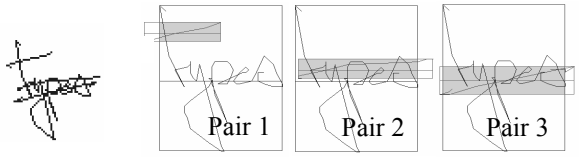



**Figure 7**. Pairs of strokes gathering.

### 3.2. Feature extraction

Eleven features are extracted from each pair of strokes in order to characterize the intersection distribution between strokes (quantity, density, dispersion), or the relative positions of the two strokes (distances between barycenters, horizontal and surface recovery), and some more global features (fig. 8).

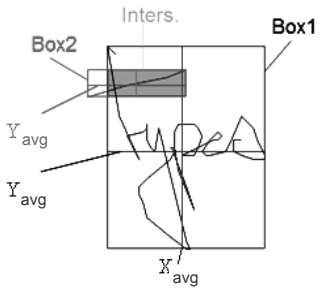



**Figure 8**. Bounding boxes with barycenters.

The eleven features used in this experiment are:

► $C1 \ldots C4$ : Characterization of the relative positions of strokes.

$$C1 = \frac{Width\ (Box\ 2)}{Width\ (Inters\ )} \qquad C2 = \frac{Surface\ \ (Box\ 1)}{Surface\ \ (Inters\ )}$$

$$C3 = \frac{|Yavg(Box1) - Yavg(Box2)|}{Hight(Inters.)} \qquad C4 = \frac{|Xavg(Box1) - Xavg(Box2)|}{Width(Inters.)}$$

► $C5 \ldots C9$ : Intersection characterization.

*C*5 : *Number of intersections between the second stroke and the other strokes.*

*C*6 : *Number of intersections between the two strokes of the pair.*

*C*7 & *C*8 : *Mean and variance of the distribution of the x-coordinates of the intersections.*

$$C9 = \frac{Nbr\ \ of\ \ interse\ \ ctions}{Length\ \ of\ \ the\ \ se\,cond\ \ stroke}$$

► *C*10 & *C*11: Global features.

*C*10 : *Number of strokes connected with the second stroke.*

*C*11 : *Deployed length of the second stroke.*

### 3.3. Decision

Pairs of strokes are classified by a multi layer perceptron (MLP), assigning to each pair of strokes its score of membership to each of the 19 subclasses. We proceeded to a cost sensitive learning of the classifier in order to keep a good distinction level between the two original classes. A cost matrix is thus introduced, which minimizes the errors of classification between subclasses of the same group, the two groups being "erasure" and "standard writing". Our MLP produces a correct classification among the 19 classes for 75.01% of the cases, with cross validation. The total scores of the classes "erasures" and "standard writing" are obtained by summing the scores of their associated subclasses. Then, the score of the "standard writing" class is simply compared to a threshold to provide the final decision, according to the following rule:

"***If*** $score_{std\_writing} < threshold$ ***then*** *decision = erasure* ***else*** *decision = standard writing* ".

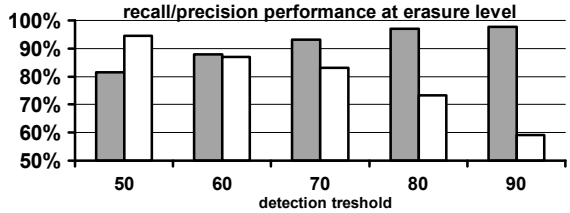
As this threshold is tuneable, it then allows acting on the system tolerance.

Once the overall decision is provided, the scores of subclasses are considered once again to divide the "erasure" class into two subclasses: "suppressive erasures" and "overwritings", thus giving the opportunity to differentiate the cases of correction.

Although the data handled by classification are pairs of strokes, the classification results on this level would badly attest the system behaviour. As an erasure could be made up of many pairs of strokes, an error of classification on one couple of strokes does not inevitably imply an undetected erasure event. Therefore, in order to get a better measure of performance that reflects these events, we split the couples of erasing strokes taking part in the same correcting event. A total score is then affected to these new entities, that can also be qualified as "suppressive erasure" or "overwriting" if the score difference is significant enough. Let us note that it is preferable to minimize the erroneous classifications as "suppressive erasures", since they could imply an definitive deletion of the concerned strokes.

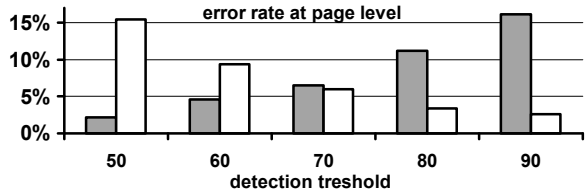
## 4. Results

For our test, 528 pages of real documents were analysed. 117 of these pages include one or more erasures. In this 117 pages, 444 erasures were listed. In our context, documents are scanned before being transmitted to the handwriting recognition process. Pages which are identified as not containing any erasure are transmitted without any additive operation, while the others are temporarily isolated in order to be subjected to a suitable processing beforehand. It is thus useful to consider the error rate on the page level (fig. 10) in addition to the recognition results at the erasure level (fig. 9).




**Recall** : $\frac{Nbr(Right\_detection)}{Nbr(Real\_erasure)}$ **Precision** : $\frac{Nbr(Right\_detection)}{Nbr(Detected\_erasure)}$

**Figure 9**. Detection results at erasure level.




**False alarm** : Erasure-free page, isolated for additive processing before handwriting recognition.

**Miss detection** : Page containing erasures, directly transmitted to handwriting recognition.

**Figure 10**. Detection results at page level.

At the time of result analysis, the manual identification of the erasures requires human interpretation, in order to firmly define the corrective events present in each page, before detection. Ideally, it would be necessary that the system locates only the corrections which will be problematic at the recognition time, and this, independently of the recognition engine. The best solution is then to act on the recognition threshold according to the behaviour of the handwritten recognition engine, until obtaining a good compromise.

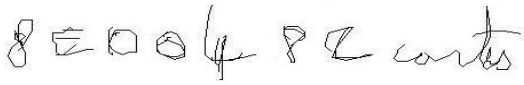


**Figure 11**. Unobvious cases.

**Figure 12**. Obvious cases.

Several tendencies appear in the erasures collected in the database. Some erasures result from a patent action of the writer. Most of these cases are not problematic for the detection process as the corrective action appears obviously on the page. Each of those obvious cases is easily detected with the minimal recognition threshold (fig. 12). Other observations show less obvious corrections done along the writing process. These must result from a writing reflex, or simply from a jerked or hesitant writing style, where the characters are regularly altered and overloaded (fig. 11). In this context, the corrections are sometimes uncluttered too much to be well distinguished from certain standard writing configurations. For too low values of the recognition threshold, the system shows a random behaviour on this type of corrections. According to the robustness of the recognition engine, it could be sometimes necessary to locate these unobvious corrections. It will then be better to increase the recognition threshold, at the price of an increased quantity of false alarms.

## 5. Conclusion

We proposed a methodology dedicated to error minimization in recognition of on-line digitized document, and in contexts that favour the presence of erasures and corrections. Our system is able to detect and locate the problematic groups of strokes while voting on their corrective nature. Our study highlights that a strict border between standard writing and erasure is hard to define objectively. Ideally, the system should anticipate all the corrections that would lead to an erroneous behaviour of the recognition engine, while discarding those non problematic ones.

The presented approach is a flexible pre-processing system dedicated to erasure localization but acting like a real recognition engine that labels also the type of erasure. Even if the erasure characterisation is based on a simple feature extraction process, the detection results of this first approach are however quite encouraging.

As future work, it could be beneficial to improve categorization of the erasures cases, by analyzing the misclassification within subclasses. A new manual labelling work could then be carried out in better conditions. We could therefore benefit from a system with a higher level of interpretation, that allows, not only to locate erasures, but also to preserve the effective content of the document by removing efficiently these problematic strokes.

## References


[1] F. Alimoglu, E. Alpaydin, "Combining multiple representations and classifiers for pen-based handwritten digit recognition", *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, Ulm, Germany, 1997, pp. 637-640.

[2] E. Anquetil, H. Bouchereau. "Integration of an on-line handwriting recognition system in a smart phone device", *Sixteenth IAPR International Conference on Pattern Recognition*, Quebec City, Canada, 2002, pp. 192-195.

[3] C.C. Tappert, C.Y. Suen, T. Wakahara. "The state of the art in on-line handwriting recognition", *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1990, vol. 12, pp. 787-808.

[4] C.C. Tappert, C.Y. Suen, T. Wakahara. "The state of the art in on-line handwriting recognition". *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1990, vol. 12, pp. 787-808.

[5] J. Hu, M. K. Brown, W. Turin. "Handwriting recognition with hidden Markov models and grammatical constraints". *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, 1994, Taipei, Taiwan, pp. 195-205.

[6] G. Seni, J. Seybold: "Diacritical processing for unconstrained online handwriting recognition using a forward search". *International Journal on Document Analysis and Recognition*, 1999, vol. 2, nbr. 1, pp. 24-29.

[7] J. Hu, M. K. Brown. "On-line handwriting recognition with constrained n-best decoding". *Proceedings of the International Conference on Pattern Recognition*, 1996, Vienna, Austria, vol. 3, pp. 23-27.

[8] S. Manke, M. Finke, A. Waibel, "Combining Bitmaps with Dynamic Writing Information for On-Line Handwriting Recognition", *Proceedings of the International Conference on Pattern Recognition*, 1994, Jerusalem, pp. 596-598.

[9] R. Plamondon, S.N. Srihari. "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, vol. 22, nbr. 1, pp. 63-84.

[10] W. Hürst, J. Yang, A. Waibel. "Error Repair in Human Handwriting - An Intelligent User Interface for Automatic On-Line Handwriting Recognition", *IEEE International Joint Symposia on Intelligence and Systems*, 1998, pp. 389-395.

[11] *Anoto*® website : www.anoto.com