Cursive Word Recognition: Methods and Strategies

Eric Lecolinet† and Olivier Baret‡

† E.N.S. Telecom Paris, 46 rue Barrault, 75013 Paris, France.
‡ A.2.I.A., 10 rue de l'Université, 75007 Paris, France.

1. Introduction

Although researchers have been working on the field of cursive script recognition (CSR) for more than thirty years, existing systems are still limited to restricted applications and this field of research remains quite open. This paper aims to present the basic principles of the techniques used so far and to classify them according to the type of strategy they are based on. It will mainly focus on off-line recognition although most of the strategies presented here are also valid for on-line recognition.

1.1. On-line and Off-line Recognition

CSR applications fall into two categories: on-line systems and off-line systems. Cursive script recognition is used in two different kinds of applications: on-line and off-line recognition systems. In the first case, handwriting is produced by using a special device (like an electronic stylus on a notepad) while off-line recognition deals with digitized images (produced by a scanner or a camera). Off-line recognition can be considered as the most general case: no special device is required for writing and "signal" interpretation is independent from signal generation, like in human recognition.

On-line recognition presents several interesting characteristics. First, recognition is performed on one-dimensional data instead of two-dimensional images. On-line data is one-dimensional in nature while off-line recognition deals with two-dimensional images. The writing line is represented by a sequence of dots whose location is function of time. This has several important consequences:

— the writing order is available and can be used by the recognition process,

— temporal information, like velocity can also be taken into account,

— the writing line has no width: input data constitutes a kind of "ordered skeleton".

Additionally, penlifts can also help recognition.

Two main philosophies have been applied to on-line recognition. The first one is *analysis-by-synthesis*, also called recognition by generation. This theory, which was first introduced by Mermelstein & Eden [71] bases recognition on the *modeling* of handwriting generation. This field nowadays constitutes a very important area of research but which is without the scope of the present paper (methods only specific to on-line recognition will not be considered in this survey). Major research in this field has been done by Plamondon & al. [82] [2], Morasso, Schomaker, Teulings & al. [106] [108] etc ...

The second philosophy consists in applying off-line methods to on-line recognition systems. In this case, temporal information is not taken into account except for ordering strokes in some systems. Although feature extraction is performed in a different way and some specificities are introduced in the recognition scheme, such systems follow overall strategies which remain similar to those implemented in off-line recognition. Thus, in addition to off-line methods, similar on-line strategies will also be considered in this paper.

Recently, several methods have been proposed that extract temporal information from static off-line data [19] [38] [8]. In section 1.4, we will discuss those methods.

1.2. The Influence of the Size of the Vocabulary and of the Number of Writers

Machine recognition being obviously far less efficient than human recognition, a tradeoff has to be found between the size of the vocabulary and the number of writers. This tradeoff greatly depends on the field of application:

— as they are mainly designed for personal computers, notepads and other dedicated interacting devices, on-line systems may limit recognition capability to a few users. Recognition is then said to be *mono-* or *multi-scriptor*. In compensation, the size of the vocabulary can be relatively large: up to 20 000 words for certain commercial softwares. An intermediate approach is also proposed by some systems: recognition is basically *omni-scriptor*, but can be more specifically "tuned" to the writing of a few frequent users.

— on the contrary, off-line systems usually require *omni-scriptor* recognition, that is to say the ability to recognize any style of writing without prior word training. This is especially true for postal or banking applications (like mail sorting or check reading) which constitute most of the existing systems or research themes. In this case, constraints are laid on the size of the vocabulary which cannot exceed a few hundred words (up to 1000 words for postal applications, but only a few tens for check reading).

1.3. Categories of Handwriting

Another major constraint is the category of handwriting that recognition systems are able to handle. Constraints can concern the quality of the writing (i.e. careful handwriting versus unconstrained handwriting) but also the style. Tappert proposed five categories to classify handwriting:

— boxed discrete characters

— spaced discrete characters

— run-on discretely written characters

— pure cursive script writing

— mixed cursive and discrete.

Many on-line commercial systems already available only deal with the first two types of writing. But recent systems are now designed for the recognition of mixed cursive writing (i.e. of type 5). This is especially true for off-line systems because of the types of applications which are considered (as in the previous subsection). This also the subject this paper will mainly focus on.

1.4. Pre-Processing and Feature Extraction

For off-line systems, available pre-processings are: binarization, noise filtering, skew correction and slant correction. Skew and slant correction rely on the evaluation of the average slope of horizontal/vertical lines. This slope can be estimated on the vertical parts of a contour-description, after a Hough Transform or directional histograms. For on-line systems, polygonization or resampling are common.

For many systems, line information is required. Some methods only look for a sequence of straight segments, others require a graph representation. Lines can be extracted by means of a global Hough transform, or through skeletonization. New methods have also been proposed on that topic, such as [94], [77].

Some new approaches have been proposed, that extract temporal information from off-line data. All of them define heuristics about handwriting such as smooth continuation of stroke direction and curvature across intersection points. A stroke graph representation is thus necessary.

Doermann and Rosenfeld [19] use a gray scale off-line image and a stroke graph built with an edge cross-section method analog to the one proposed in [102]. Local clues are related to ink deposit variations which may reflect the direction of a stroke. Regional clues involve heuristics about the writing process such as smooth changing of curvature across a junction point. Global clues involve consistency constraints on the whole graph. This "Stroke Recovery Platform" has been used for overlapping lines separation in [20].

Govindaraju and al [38] use a skeletonized binary image, which is equivalent to a stroke graph. Heuristics are applied on the skeleton in order to segment it into ordered strokes. Character templates are proposed, based on a temporal description of strokes.

Boccignone and al [8] also start from a skeleton which has been smoothed with polygonization. Good continuity criterion is combined with width variation of stroke. Variations in length of polygonal segments along a stroke are also taken into account.

1.5. Organization of the Next Sections

The next section contains a discussion about the strategies used for CSR, and their adaptation to different kinds of problems. Section 3 will present the major methods and techniques developed for cursive script recognition. This section is itself divided into three subsections corresponding to the different categories of strategies previously described. Hybrid systems combining different strategies will be considered in section 4. Finally, section 5 will present available results and will then come the conclusion.

2. Strategies

Two main types of strategies have been applied to the CSR problem since the beginning of research in this field: the holistic approach and the analytical approach. In the first case recognition is globally performed on the whole representation of words and there is no attempt to identify characters individually. In the second case, words are not considered as a whole, but as sequences of smaller size units. Recognition is not directly performed at word level but at an intermediate level dealing with these units which are usually more or less closely related to characters. These units can be of different kinds according to the method, like: graphemes, segments, pseudo-letters, etc ...

As they rely on the recognition of intermediate word units, analytical methods are also said to be segmentation based. In this case, "segmentation" refers to "letter" or "word unit" segmentation, that is to say some kind of intermediate unit that is considered as a whole by a recognizing step and which has to be separated from the rest of the word. However, the word "segmentation" can be confusing as all pattern recognition methods basically perform some kind of segmentation. For instance, holistic methods usually perform feature extraction, which can also be considered to be segmentation stage. The main difference lies in the level of abstraction of the segmented elements: features (that is to say low level elements) in the case of holistic methods, versus pseudo-letters in the case of analytical methods. This theoretical difference is also significant from the practical point of view as it can also lead to very different schemes with different properties and performances (as will be seen in further sections).

2.1. Holistic Strategies

Holistic methods follow a two-step scheme:

— the first step performs feature extraction,

— the second step performs global recognition by comparing the representation of the unknown word with those of the references stored in the lexicon.

This scheme leads to two important practical consequences:

- as letter segmentation is avoided and recognition performed in a global way, these methods are usually considered to be tolerant to the dramatic deformations that affect unconstrained cursive scripts.
- as they do not deal directly with letters but only with words, recognition is necessarily constrained to a specific lexicon of words.

The second point is especially critical when training on word samples is required. In this case, the lexicon cannot be automatically updated from letter information. A training stage is thus mandatory to expand or modify the lexicon of possible words. This property makes this kind of method more suitable for applications where the lexicon is statically defined (and not likely to change), like check recognition. They can also be used for on-line recognition on a personal computer (or notepad), the recognition algorithm being then tuned to the writing of a specific user.

More general applications require a dynamic generation stage of holistic descriptions. In this case, a specific stage must be used to convert any word from its ASCII form to the holistic representation required by the considered recognition algorithm. Word representation must then be generated from generic information about letter and ligature representations using a reconstruction model. Word reconstruction is required by all applications dealing with a dynamically defined lexicon, like for instance postal applications where the list of possible city names is derived from zip code recognition. Several studies address this problem and will be considered in the section devoted to the description of holistic methods.

2.2. Analytical Strategies

Analytical strategies deal with several levels of representation corresponding to increasing levels of abstraction. Three levels of representation are usually implicitly considered: the feature level, the word level and an intermediate level dealing with subparts of the words (that we will call "word units"). The way these units are defined and extracted from the whole word is at the very core of analytical recognition. A common rule is that these units must be easily related to characters in order to make recognition independent from a specific vocabulary. Because recognition is then basically letter-based, the vocabulary can be defined dynamically (in a symbolical form which can vary) and no word training is required.

Many kinds of word units (and designations) have been proposed so far. Some methods use letters, others use graphical units called graphemes, segments or pseudo-letters. A major difference between analytical methods lies in the way word unit segmentation is performed. This makes analytical approaches fall into two main categories:

— analytical approaches with *explicit* (or *external*) segmentation,

— analytical approaches with *implicit* (or *internal*) segmentation.

In the first case, words are *explicitly* segmented into letters (or pseudo-letters) which are then recognized individually, while in the second case segmentation and recognition take place at the same time.

2.2.1. Analytical Approaches with Explicit Segmentation

The first step deals directly with pixel (or signal) information in order to detect features. Then these features are compared to class prototypes to recognize the letters contained in the word. Contextual high-level knowledge (that is to say lexical, syntactic or semantic knowledge) is then used at the end of the process to ensure proper word identification.

This category of methods includes the three following steps:

— external segmentation of the word into smaller units (segments, graphemes...),

— individual recognition of these units,

— contextual post-processing (CPP) using lexical, syntactic or semantic knowledge,

The main advantage of the explicit approach is that segmented units can be recognized using a classical OCR technique. Its main drawback is that erroneous segmentation may lead to incorrect recognition. Moreover, as the segmented units are recognized and labeled individually, subtle information about how they differ from the ideal references may be definitively lost instead of being efficiently used by the contextual post-processor to correct possible mistakes.

Contextual post-processing is performed at the end of the process to ensure proper word identification. The techniques used may be of two kinds:

— Orthographic correction techniques using the statistics of the vocabulary (methods mainly based on n-grams frequencies, on Markov Models and the Viterbi algorithm)

— Direct comparison with a lexicon (methods using algorithms based on Dynamic Programming like Edit Distance, DP-matching, etc ..)

In theory, as no lexicon is used in the first case, any word of the considered language can be recognized, while recognition is limited to the words contained in the lexicon in the second case. However, the drawback of statistical methods is that they may possibly produce non existing words. A dictionary lookup is thus necessary and alternative choices have to be considered in case of failure. But still, these techniques can save much computation time for large vocabularies.

A classical remark is that explicit methods lead to the famous paradox according to which ‘‘it is necessary to segment to recognize, but it is also necessary to recognize to segment''. However these approaches may be justified by visual evidence: objective segmentation marks (like ligatures between letters) really exist in cursive scripts and make the reading process easier, even for humans. According to the principles of perceptual organization and the Gestalt laws, most of the characters should be segmentable without recognition. Thus, these systems usually follow a "double step" segmentation strategy:

— first, the segmentation step explicitly divides the words into smaller units close to characters,

— then, final "segmentation" is performed implicitly at the contextual stage taking simultaneously into account the possible relationships between segments and characters and the lexical constraints of the vocabulary.

2.2.2. Analytical Approaches with Implicit Segmentation

Such methods also segment words into individual units (which are usually letters). But the main difference is that they perform recognition-based segmentation: letter segmentation is then a by-product of letter recognition. The main interest of this category of methods is that they bypass the segmentation problem: no complex segmentation algorithm has to be built and no recognition error may occur because of incorrect segmentation. The basic principle of analytical methods is derived from [14] where a mobile window of variable size is used to provide "tentative segmentations" which are confirmed or not by character recognition.

Word level knowledge can be brought in during the recognition-segmentation stage. This contextual knowledge can either be introduced in a statistical way, by using a lexicon, or even in both ways simultaneously.

Statistical representation has become very popular these last few years with the use of Hidden Markov Models. In these models, contextual information is represented by means of transitional letter probabilities. A specific subsection will be devoted to this important field of research.

2.3. Top-down Versus Bottom-up Strategies

Another point of view is to consider whether these strategies follow a bottom-up or a top-down recognition scheme [92]. Because they perform recognition by a direct comparison with the lexicon of descriptions (without including any intermediate recognition stage) holistic methods have often been considered as following a top-down strategy. Conversely, as they often deal with hierarchical representations (features, letters, words) processed at successive stages, analytical methods are usually seen as bottom-up strategies.

However, holistic methods still incorporate a bottom-up phase as they transform the initial word image (or signal) into a set of features, while some analytic methods deal simultaneously with segmentation, recognition and contextual analysis. So, it can be said that most techniques used today are somewhat hybrid and do not strictly fall into any of these two categories. We will thus consider holistic methods as being "mostly" top-down" and analytical methods as being "mostly" bottom-up.

However, "pure" top-down approaches have also been proposed ([67] [63]) and will be considered in a specific section.

3. Methods and Techniques

3.1. Holistic Methods

3.1.1. Early Studies

Whole word recognition was introduced by Earnest [23] in the beginning of the sixties. Although it was designed for on-line recognition, his method followed an off-line methodology: data was gathered by means of a "photo-style" in a binary matrix (no temporal information was used). The method was based on a comparison of a collection of simple features extracted from the whole word with a lexicon of "codes" representing the "theoretical" shape of the possible words. Feature extraction was based on the determination of the "middle zone" of the words (that is to say the horizontal area that does not contain any vertical extension): ascenders and descenders were found by considering the part of the writing exceeding this zone and internal loops were detected in the included part. This global representation was then compared to those of the lexicon using a transcoding table describing all the usual ways of writing letters.

Another on-line technique proposed by Miller [72] in 1971 was based on a decomposition of the words into macro-feature segments. These segments constituted a set of basic writing units whose combination was assumed to describe the shape of any existing word. They were obtained by shape analysis and were not related to characters (for instance a single feature could be part of several characters). The segments were then encoded to a value representing the successive directions of their contour and angular distance was used for global comparison with the references of the lexicon.

Brown and Ganapathy [11] used features vectors to represent word characteristics in a global way. All the extracted features (which consist in extensions, cups, strokes ...) are related to two sets of overlapping windows which divide the word into equal parts. The number of windows is derived from an estimate of the number of characters contained in the word. Classification is then done on these feature vectors using K-Nearest Neighbors.

3.1.2. Hints, Propositions and Remarks

These three methods are interesting because they introduce several techniques which are still used by current methods (even if calculations are now done in a more sophisticated way). For instance, many techniques still use middle zone determination to detect the ascenders and descenders of the words. The type of features extracted also remains globally the same: ascenders, descenders, directional strokes, cusps, diacritical marks, etc ... Angular differences are also still used for on-line recognition.

On the other hand, the main drawback of these early techniques (and especially the first two ones) lies in the the way they perform comparison between hypotheses and references. The techniques used were not flexible enough and were generally unable to take into account strong deformations of the writing (or segmentation errors in the second case).

More efficient techniques, which were originally introduced for speech recognition are now commonly used. These techniques are based on Dynamic Programming and essentially fall into two categories:

— methods based on distance measurements

— methods based on a probabilistic framework.

The first type of method is based on Edit Distance, DP-matching or similar algorithms while the second one uses Markov or Hidden Markov Chains.

3.1.3. Dynamic Programming and Edit Distance

Dynamic Programming (DP) is a general technique for solving optimization problems. It has been used for many applications in the field of pattern recognition and especially in speech and character recognition. The methods presented in this subsection are based on a specific algorithm called Edit Distance (ED) which was introduced by Wagner and Fischer [109] for speech recognition. Note that other important algorithms, like the Viterbi algorithm [31] which is used to process Markov Chains, are also based on Dynamic Programming.

Edit Distance is a matching algorithm which evaluates similarity (by computing a "distance") between two strings of symbols. Given a set of elementary transformations between the symbols of the two strings (and an associated cost), this algorithm provides the best sequence of elementary transformations necessary to transform the first string into the second one. The cost associated with this optimal transformation gives the "Edit Distance" between the two strings. Moreover, Wagner and Fischer proved that ED defined "real" distance (and was really optimal) under certain assumptions. Many variants (or earlier versions) of this algorithm have been studied like the Levenshtein distance or the DP-matching or "non-linear-elastic-matching" techniques which are suboptimal algorithms.

Given two strings X = X1 X2 ... Xn and Y = Y1 Y2 ... Ym, Edit Distance was originally based on the three elementary transformations:

— substitution of a symbol Xi into Yj with the associated cost d(i,j)

— deletion of a symbol Xi with the associated cost d(i,@) ("@" being the null symbol)

— insertion of a symbol Yj with the associated cost d(@,j)

ED is then computed using the recursive formula (ED(i,j) represents the distance between substrings X1..Xi and Y1...Yj):

$$ED(i,j) = \min \begin{cases} ED(i-1,j-1) + d(i,j) \\ ED(i-1,j) + d(i,@) \\ ED(i,j-1) + d(@,j) \end{cases}$$

$$\text{with } ED(0,0) = 0$$

Such a technique was employed by Moreau & al. for a holistic recognition method which was part of a larger system [74][84]. Words were represented globally by a list of features indicating the presence of ascenders, descenders, directional strokes and closed loops. In order to take into account y-location, the word was divided into three horizontal zones separated by two guidelines which were not mere straight lines, but smooth curves following the central part of the word, even if it was slanted or irregular in size. Relative y-location (to the guidelines) was associated to every feature and uncertainty coefficients were introduced to make this representation more tolerant to distorsion by avoiding binary decisions. Finally, matching was performed globally on the dictionary of encoded references by means of Edit Distance. Variants of this technique have been included in two hybrid systems, one for check recognition and another one for city name identification.

Similar systems were also developed by Paquet & Lecourtier [79] and by Leroux and al. [65] for check recognition. Final decision was based on Dynamic Programming in both methods but the features used were different. In the first case they were based on the notion of "guiding points", which are defined as the intersection of the letters and the median line of the word. In the second case, features were derived from the contours of words.

Camillerapp & al. [13] used graph representation to describe cursive words. Graphs were deduced from a skeleton. Horizontal and vertical locations were also taken into account and two-dimensional tree-to-tree comparison was eventually performed using Dynamic Programming.

Madhvanath and Govindaraju recently designed a holistic method for lexicon filtering [69]. The aim of this method is not to find "the best solution" but to reduce the size of the lexicon (which is then processed using another technique). Representation is based on *wordgraphs*, whose nodes correspond to feature attributes, an associated scalar or positional value and a confidence measurement. Lexicon words (which share the same representation) are "predicted" by using heuristic rules. Constrained bipartite matching is performed using a possible association matrix. Entries of the lexicon are ranked by combining three measures of goodness called: Confidence of Match, Closeness of Match and Degree of Mismatch. The system achieves 50% size reduction with under 2% error.

3.1.4. Markov Models and Probabilistic Methods

One of the first CSR systems using Markov Models was developed by Farag [27] in 1979. In this method (which was designed for on-line recognition), each word is seen as a sequence of oriented strokes obtained by using a graphic tablet. All strokes are of similar length and are represented using the Freeman code. The model of representation is a non stationary Markov Chain of the first or second order. Each word of the lexicon is represented as a list of stochastic transition matrixes. Each matrix corresponds to an interval of time and contains the transition probabilities from the (j)th stroke to the following one. The eight directions constitute the states of the Markov Chain and the initial matrix contains the a priori probabilities of the first stroke of the word. The recognized word is the reference Wi of the lexicon which maximizes the joint probability P(Z,Wi) where Z is the unknown word. Using Bayes formula, this is equivalent to multiplying a priori probability P(Wi) by conditional probability P(Z/Wi) which is also the product of the corresponding elements of the transition matrixes computed for reference word Wi.

More recently, Hidden Markov Models (HMM) became very popular in CSR. Nag, Wong and Fallside [75] used this technique for the recognition of literal digits. This system was also on-line and features were represented using an angular representation. Many other methods are based on HMMs, but are classified (an described) in the section devoted to analytical methods with implicit segmentation.

Gilloux and Leroux [34] use Markov Chains for off-line cheque recognition. Several Markov models are used at different recognition stages (word recognition and cheque amount recognition). Context is taken into account via prior probabilities of words and word trigrams.

Another method for the recognition of noisy images of isolated words such as in cheques was recently proposed by Gorsky in [37]. In the learning stage, lines are extracted from binary images of words and accumulated in prototypes called *holographs*. The technique used for line extraction has been presented in [94]. One prototype is used for each class of words. The vocabulary is fixed and small (less than 30 different words). During the test phase, correlation is used to obtain a distance between an unknown word and each word prototype. Using these distances, each candidate word can be represented in the prototype space. Each class is approximated with a Gaussian density inside this space. For the word under recognition, the probability that it belongs to each class is calculated using those densities. Other simple holistic features (ascenders and descenders, loops, length of the word) are also used and combined with this main method.

Lastly, Houle presented a holistic approach based on Collective Learning Systems (CLS) theory [50]. In this method, a set of fuzzy features is extracted from an observation window which is slided along the word to train a collective learning automaton. This observation window is centered around the successive strokes of the word (which are extracted by analyzing the curvature of the contour) in order to produce a stream of overlapping samples. This method was initially used to discriminate between "Detroit" and "non-Detroit" words taken from real mail.

3.2. Analytical Methods with Explicit Segmentation

Analytical recognition has quite a long history: it was already studied by Frishkopf and Harmon at Bell Laboratories at the very beginning of the sixties for on-line recognition. Their study is probably one of the very first attempts to tackle the difficult problem of character segmentation which is based on "landmark" extraction (like ascenders and descenders) and letter width estimation. Words are then divided into segments of similar width which are centered around the pre-detected landmarks. However, it must be noted that this scheme does not perfectly work with letters "u", "n", "m" ... as they do not contain landmarks.

Letter recognition was done using a decision tree that involved structural features such as: vertical and retrograde strokes, cusps, closures, diacritical marks. The vertical location of these features was also taken into consideration by separating the word into three horizontal zones: the middle, upper and lower zones.

As the authors noticed that many character combinations were not legible, (they report that 42% of all possible bigrams do not appear in any English word, and 50% have a probability which is lower than 0.0003), they eventually added a post-processing phase to the recognition system. This post-processor used bigram statistics to detect errors and trigram statistics to correct them.

In 1973 Sayre published an interesting study which was probably the first off-line system for cursive script recognition. Segmentation was based on the detection of characteristic areas of the drawing and the contextual phase was also based on bigram and trigram analysis. An interesting idea is that the recognized classes of the classifier do not correspond to letters but to the *possible shapes* of letters. Thus, only 17 *non-exclusive* classes were considered, so that several letters can correspond to the same class and vice-versa.

The critical phases of all these methods are obviously segmentation and contextual analysis. Next subsections report a few techniques which have been developed to improve them both. Recognition will not be dealt with as it can be solved by classical OCR or CSR techniques.

3.2.1. Character Segmentation

Techniques for segmenting cursive script are all based on heuristic rules derived from visual observation. Obviously, there is no "magic" rule and it is not possible to segment any handwritten word into perfectly separated characters in the absence of recognition. Thus, word units resulting from segmentation are not only expected to be entire characters, but also parts or combinations of characters. These segmented units are usually called "graphemes" (or "segments" or "pseudo-characters"). It must be noted that the relationship between characters and graphemes must remain simple enough to allow an efficient post-processing stage. In practice, this means that a character must not be divided into more than two graphemes and vice-versa in most cases.

As said before, these methods can be seen as following a double segmentation strategy: first *actual* segmentation points or boundaries are found which are then *implicitly* modified during the contextual (or recognition) stage. Thus, this kind of segmentation is often called "pre-segmentation" or "loose-segmentation", while resulting segmentation points are said to be "possible segmentation points" (PSP).

Most segmentation techniques are based on the fact that most lowercase characters are linked together by *lower ligatures*. So, an easy way to locate ligatures is to detect the minima of the upper outline of words. Unfortunately, this technique has several drawbacks which must be solved in an appropriate way:

— letters "o", "v" and "w" are usually followed by "upper" ligatures,

— letters "u", "w" .. contain "intra-letter ligatures", that is to say that a subpart of these letter cannot be differentiated from a ligature in the absence of context

— artifacts can sometimes cause erroneous segmentation.

Many of these problems can easily be solved at the contextual stage (and especially those of the second type because the drawing is ambiguous in nature without the help of lexical context). However, the quality of segmentation still remains very much dependent on the design of the implemented segmenter.

Segmentation techniques based on the above principle were for instance developed by Ehrich & Koelher [24], Maier [70] and Lecolinet [61] [62]. This last study was based on a dual approach:

— the detection of possible pre-segmentation zones,

— the use of a *pre-recognition* algorithm, whose aim was not to recognize characters but just to evaluate if a piece a drawing was likely to constitute a valid character.

Pre-segmentation zones were detected by analyzing the upper and lower profiles and the opened concavities of the words. Temptative segmentation paths were then created to separate cursive words into isolated graphemes. These paths were required to respect several heuristic rules dealing with continuity and connectivity constraints. Finally, pre-segmentations were only validated if coherent with the decisions of the pre-recognition algorithm. Hierarchical segmentation was also performed in some special cases. An important property of this method was to be mostly independent to character slant, so that no special pre-processing was required.

The same goal was achieved by Holt & al. [49] by analysis of the upper contour and a set of rules based on contour direction, closure detection, and zone location. Upper contour analysis was also used by Kimura, Shridhar & Narasimhamurthi for a pre-segmentation algorithm which was part of the second stage of an hybrid recognition system [59]. Moreover, in the first stage of the same system, they also implemented another segmentation technique based on a "hit and deflect strategy" (HDS). This interesting technique was initially developed for the segmentation of digit strings [93].

Another interesting technique for segmenting handwritten strings was also proposed by Fenrich in [30]. This method, which is able to cope with strings of variable length, proposes a recognition aided approach. It is also based on upper and lower contour analysis and on a splitting technique inspired from HDS.

A few other techniques use different criteria like those of Bozinovic and Srihari where segmentation is based on the detection of minima of the *lower* contour [10]. Presegmentation points (PSP) are found around these locations and emergency segmentation is performed when PSPs are too far from each other. The main drawback of this method is that it requires handwriting to be previously deslanted to ensure proper separation.

In a recent study which aims to locate "key letters" in cursive words Cheriet employed background analysis to perform letter segmentation [18]. In this method, segmentation is based on the detection and the analysis of the faceup and facedown valleys and open loop regions of the word image.

Higgins and Ford designed a two stage algorithm for on-line recognition [46] (all previous techniques being more specially devoted to off-line segmentation). First, possible segmentation points were generated by detecting specific features of the pen-strokes of the script (like cusps, intersections, points of inflection and end points). Then, irrelevant PSPs were eliminated during a second stage based on segment analysis.

Finally, we want to mention that a number of methods for segmenting on-line cursive script into allographs were reviewed by Teulings and Schomaker [107] (a method of their own based on simulated annealing was also presented in the same paper) and that a recent survey of segmentation techniques was also published by Dunn and Wang [22].

3.2.2. Contextual Post-Processing

In [9], Bozinovic and Srihari presented a string correction algorithm for cursive script recognition which was derived from research on speech recognition [3]. Unlike older techniques such as the Viterbi algorithm [31] this method is not only able to handle substitution errors but also splitting and merging errors (although in an approximate way in the second case). A channel model for CSR is represented by a finite state machine allowing such operations. The best estimate is obtained by using a stack decoding algorithm (a technique derived from the "branch and bound" concept of search) combined with a *trie* structure representation of a dictionary. A trie is a tree representation of a lexicon where nodes represent letters and leaves represent the words which are made of all the encountered (letter) nodes from the root to each considered leaf. This data structure is of great interest for word recognition and has been widely used in that field.

A few years later, the same authors described a whole system based on an adaptation on this previous idea [10]. Their model followed an analytical bottom-up scheme including pre-processing, pre-segmentation, feature extraction and letter and word hypothesization. These words were hypothesized using a modified version of the stack decoding algorithm by storing the most likely prefixes and expanding them until the end of the word was reached. Note that the pre-segmentation stage of this method is described in the previous section.

Lecolinet and coworkers used Extended Edit Distance in a method which was applied to check and city name recognition [61][62]. Grapheme Recognition was either performed using a statistical algorithm (based on linear separation of feature vectors derived from concavity detection) or by using a multi-layered neural network grapheme images. An extended version of the Edit Distance making use of eight elementary transformations (instead of three in classical ED) was implemented to model properly various cases of merging, splitting and contextual substitution of graphemes. In a recent version of the method, automatic learning capability was added.

In [60] Kundu and coworkers propose the use of Hidden Markov Models as a contextual post-processor. In this paper, graphemes were supposed to have been segmented by another technique. More recent papers by the same authors employ implicit HMM systems which will be described in the next section.

Finally, it should be noted that a review of segmentation and contextual analysis techniques for text recognition was published by Elliman and Lancaster in [25].

3.3. Analytical Methods with Implicit Segmentation

Implicit methods have been the subject of great attention during these last few years because they combine most of the advantages of holistic and analytical strategies:

— no complex segmentation algorithm has to be built; this scheme allows one to bypass segmentation problems,

— as they are letter-based, these methods are not limited to a specific lexicon: they can easily be applied to the recognition of any vocabulary,

— the identification of the word is performed globally, segmentation being a consequence of recognition which is itself driven by contextual analysis.

Depending on the type of technique used, recognition can either be performed by following a local or global optimization scheme. In the first case, recognition is done iteratively in a left-to-right scan of words. Conceptually, these methods are derived from a scheme developed by Casey and Nagy [14] for the recognition of handprinted words. The basic principle of their method was to use a mobile window of variable width to provide "tentative segmentations" which were confirmed (or not) by character recognition.

Other methods proceed in a more global way by generating a lattice of all (or many) possible feature-to-letter combinations where final decision is eventually taken by considering the whole graph. Optimization is then performed globally to bypass possible errors occuring on the beginning of the word.

Recent years have seen increasing interest in Hidden Markov Models (HMMs). This technique leads to very interesting and efficient systems, most of which can be considered as following an analytic implicit strategy. The next section will be devoted to their description. However, many other interesting schemes have also been proposed and are the object of the following paragraphs.

3.4. Non-HMM Implicit Methods

Peleg and Hayes [81] [43] used *probabilistic relaxation* to read off-line handwritten words. Hayés model was working on a hierarchical description of words derived from a skeletal representation. Relaxation was performed on the nodes of a stroke graph and of a letter graph where all possible segmentations were kept. Complexity was progressively reduced by keeping only the most likely solutions. N-gram statistics were also introduced to discard unlegible combinations of letters. A major drawback of such techniques is that they require intensive computation.

Tappert employed *Elastic Matching* to match the drawing of the unknown cursive word with the possible sequences of letter prototypes [103]. As it was an on-line method, the unknown word was represented by means of the angles and y-location of the strokes joining digitization points. Matching was considered as a path optimization problem in a lattice where the sum of distance between these word features and the sequences of letter prototypes had to be minimized. Dynamic programming was used with a warping function allowing to skip unnecessary features. Digram statistics and segmentation constraints were also eventually added to improve performance (the purpose of segmentation constraints was both to avoid certain errors and to speed up the process by reducing the number of prototype comparisons).

Berthod and Ahyan developed a structural system involving two complementary notions: the *aspect* and the *structure* of words [6][7]. "Aspect" characterizes the profile of the words and is deduced from ascender and descender detection. "Structure" was defined as a sequence of structural features (like x- and y-extrema, curvature signs, cusps, crossings, penlifts, and closures) collected all along the word. This sequence was used to generate all the legible sequences of letters that would have this "structural" representation. This was done by means of a tree dictionary containing the description of letter scripts (previously learned during the training phase). Linking features (corresponding to ligatures) could be added between letters. "Aspect" was then taken into account to chose the best solution(s) among the list of generated words. Thus, this method was also following a hypothesis testing and verification scheme, an interesting strategy which was first experimented by Hull for handprinted word recognition (see for instance [53]).

A structural approach following a lexicon-based strategy was also proposed by Lorette et Ouladj for on-line recognition [78]. Words and letters were represented by means of two tree dictionaries: a letter tree† in the first case and a feature tree in the second case. These features consisted in 4 directional codes and in ascenders or descender markers. Recognition was done by means of a prediction / verification strategy: letters were predicted by finding in the letter tree the paths compatible with the extracted features and they were verified by checking their compatibility with the

† this type of data structure is also called a "trie" and is described in section 3.2.

word dictionary.

The system of Higgins and Whitrow [45] implemented a hierarchical description where primitive on-line features were progressively grouped into more sophisticated representations. The first level corresponded to the "turning points" of the drawing (and their associated attributes). The second level was dealing with more sophisticated features called "primary shapes". Finally, the third level was a treillis of temptative letters and ligatures. Ambiguities were then resolved by contextual analysis using letter quadgrams to reduce the number of possible words and a dictionary lookup to select the valid solution(s).

A different approach uses the concept of regularities and singularities [95] [96]. In this system, detection of the "singular parts" of the word (which are complementary of the uninformative "regular parts") produces a symbolic description chain. On a stroke graph obtained after skeletonization, the regular part is the sinusoid-like path that joins all cursive ligatures. Singular parts are stroke-trees. The most robust features and characters (the "anchors") are detected first from this description chain. Dynamic matching is then used for analyzing the remaining parts. Segmentation is thus produced implicitly.

An original technique combining dynamic programming and a neural net recognizer has recently been introduced by Burges and coworkers [12]. This technique, called Shortest Path Segmentation (SPS), selects the optimal consistent combination of cuts from a set of candidate cuts generated by using heuristics. Given this set of candidate cuts, all possible "legal" segments (which are expected to correspond to characters) are constructed by combination. A graph whose nodes represent acceptable segments is then created and these nodes are connected when they correspond to compatible neighbors. The paths of this graph represent all the legal segmentations of the word. Each node of the graph is then assigned a cost obtained by the neural net recognizer (which is a Space Displacement Neural Network). The shortest path through the graph thus corresponds to the best recognition and segmentation of the word.

Lastly, Fujizawa proposed an interesting technique called Parallel Dynamic Programming [32], which aims to process all the entries of the lexicon at the same time. The goal of such an approach is to reduce computation time in order to deal with larger lexicons. The method is based on dynamic programming and the A* algorithm.

3.5. Hidden Markov Models

3.5.1. Description

Hidden Markov Models have become a very popular technique for cursive recognition, for on-line as well as off-line systems. The basic idea is to model cursive writing as a hidden underlying structure which is not observable. This underlying structure is a canonical representation of the reference patterns (e.g. a sequence of letters). Deformations in their drawing produce observable symbols. Hence, HMMs are able to model the fact that for example different letters may have similar shapes, and that a given letter may be written in many different ways.

A discrete HMM is entirely defined by a set of states Q, a fixed vocabulary of symbols V, probabilities of initial states "pi", transition probabilities between states A and observation probabilities of symbols within states B [85].

HMMs are probabilistic models, which allows them to integrate easily knowledge coming from different other probabilistic sources. Most of the time, observations are discrete symbols extracted from a fixed vocabulary. Features within segmentation boundaries are quantified using a VQ codebook, or KNN algorithm. In some cases, observations are not discrete symbols, and observation probabilities are continuous probability densities [4]. Most HMM-based word recognition systems use an intermediary level recognition stage, by computing an observation probability at letter-state or letter-subHMM level. Hence they are analytical methods.

Context (word and letter frequencies, syntactic rules) is supported by transitional probabilities between letter states. In most applications, first order Markov models are used: transitional probabilities occur between state t and state t+1 only. [60] is an example of second order HMMs. A path through an HMM is a state sequence. Its evaluation is obtained by multiplying transitional probabilities and observation probabilities along that state sequence, i.e. joint probability of a state sequence and observation sequence, given a model.

In most cases, the evaluation of an observation sequence by a HMM is calculated by finding the best path in the model. This can be done with the Viterbi algorithm, based on a time efficient dynamic programming scheme. HMM transitional and symbol probabilities can be learned using an incremental algorithm: the Baum-Welch algorithm [85]. Starting from an initial evaluation, HMM probabilities are re-estimated using probabilities of observed symbols of the training set. A detailed survey of the learning of HMMs probabilities can be found in [36].

3.5.2. Word HMMs

For word recognition problems, HMM states may be letters or intra-letter states coming from letter-HMM models. They can also correspond to the successive positions of canonical observations.

There are two classes of Hidden Markov Models for word recognition: model discriminant and path discriminant. With model-discriminant HMMs, one model Mk is constructed for each word. Words can be ordered using the probabilistic evaluation of observation sequence O, given the model Mk [33], [17], [89], [5].

With path discriminant HMMs, [60], [15] [16] only one HMM model is constructed. Word recognition is performed by finding the k best paths through this unique model. Each path is equivalent to a sequence of letters. In case of a restricted word lexicon, invalid letter sequences have to be ignored during the search of the k best paths, or eliminated using a lexicon comparison module.

Path discriminant HMMs can handle large vocabularies, but are generally less performing (in terms of recognition rate) than model-discriminant HMMs.

3.5.3. Letter HMMs

In the case of touching discrete or unconstrained writing style, it is impossible to find perfect letter segmentations. HMMs have been used to model the fact that letters do not always fit within segmentation boundaries. Most HMMs dealing with unconstrained cursive script are built from elementary model-discriminant HMMs describing letters, which are then combined into several model discriminant word HMMs [33] [17] or a single path-discriminant model [15]

For a letter HMM model, transition probabilities can be learned by observing the letter segmentation behavior on a training set [17] [89].

Chen and al [17] use compound symbols. Letter HMM states correspond to the different numbers of individual observations (i.e. letter segmentation units) that can be matched to the letter. Thus, letters like 'm' and 'i' can have different HMM state structures. Compound symbol probabilities and time duration probabilities (i.e. the number of segmentation units for a given letter) are learned for each letter during training stage. Virtual observation probabilities and transition probabilities for letter HMM states are defined so that joint probability of observation and intra-states path given a letter HMM are equal to time duration probability multiplied by compound symbol observation probability.

Bellegarda and all [4] propose a model-discriminant HMM for on-line character recognition. Observations are small portions of writing called "frames". Several kinds of feature vectors (each kind belonging to a "chirographic space") are extracted from those frames. For each chirographic space, there is a given set of continuous probability densities within each state of the model. Letter models can be degenerate (using only one state), to provide a time-efficient character pre-recognizer.

4. Hybrid Systems

Many real-world cursive recognition systems have to cope with heterogeneous data. Variability may come from various styles of writing, a great number of scriptors, or writing material (paper or pen). Designing a single module, able to recognize so many different types of input data seems difficult or even impossible, unless identification methods are very approximate and the context very strong. Noisy versus clean data may also require adapted methods.

Such variation of input data can be solved with several specialized modules combined in order to achieve a general system. Module combination can be sequential or parallel. A sequential approach combines pre-classification algorithms, whose output is passed on to specialized recognition modules like top-down word verification procedures or writing-style specific algorithms. A parallel approach will analyze input data through competing points of view and combine them later. The combination output can also be analyzed with specialized or reduced context recognition modules.

4.1. Sequential Approaches: Pre-Classification Methods

Two main approaches have been used to build a sequential system. Some systems analyze the writing style of input data, and apply specialized recognition methods for each writing style. Others use general purpose classifiers whose output consists for example in a reduced word vocabulary.

4.1.1. Writing Style Evaluation

Favata and al [29] first segment the word images into units corresponding to discrete characters. Depending on the confidence levels resulting from a handprinted recognition module, they classify those units as discrete characters, or not. Non-discrete regions are analyzed with two modules, one designed for touching discrete, the other for cursive script. Each module has its own letter-segmentation rules. Their output consists in character recognition and possible segmentation points. They are combined at word level. Several paths are chosen among the merged possible character identifications and segmentation alternatives. The selected paths are then matched to a word lexicon and a list of possible valid solutions is produced.

Plessis and al [84] directly analyze the writing style. For this task, they estimate the dimensions and shape of connected components. Then, three different recognition modules are used. One is a global method, used only on pure cursive script. The second one is analytical, based on a statistical character recognition module. The third one is also analytical and uses a Neural Network for character recognition.

4.1.2. Reducing Context with Increasingly Detailed Diagnostics

Madhvanath and al [69] present a global handwritten word recognition method for lexicon reduction. This method has been designed to produce a robust word recognition adapted to bad quality data. They use ascenders, descenders and word length.

Kimura and Shridar [59] chain two analytical methods, one with an external segmentation scheme using fuzzy rules and edit distance, the other with an implicit and detailed segmentation algorithm.

Chen and al [17] use a two-stage HMM system for cursive words recognition. An initial large vocabulary is filtered through a path-discriminant HMM, whose output is a middle-sized vocabulary. A set of model-discriminant HMMs then produces a ranking inside this vocabulary subset. Their architecture is based on the fact that model-discriminant HMMs perform better than path-discriminant ones but become prohibitive in terms of speed, memory and learning material when the vocabulary becomes larger.

4.2. Parallel Approaches

For this type of methods, much work has been done to build classifier combination modules. Competing classifiers may provide different levels of information. Different combination shemes must thus be used depending on the type of output which is provided by the individual classifiers to be combined. The three following types of output can be provided by classifiers:

— a list of classes proposed for the candidate description [51].

— a list of classes with their rankings [48].

— a list of classes with a numerical value expressing confidence or diffidence. Most of the time, those values are probabilities.

It should be noted however that some of the methods cited here were not specifically designed for cursive script.

4.3. Methods based on a Verification Strategy

Li and al [67] present a printed word recognizer suitable for reduced vocabulary. Inside this reduced set, they compare words two by two. They have developed a dynamic feature selection algorithm to discriminate between two characters. Hence they construct specialized modules able to discriminate pairs of characters. Given a couple of words, discriminating characters are selected and processed through character discrimination modules. A distance is computed for each pair of differing characters and one of the two words is said to win the competition. Among the reduced vocabulary, words can be ranked according to the number of times they win against the other words.

Lecolinet [63] proposes a top-down directed word verification method called "backward matching". In cursive word recognition, all letters do not have the same discriminating power, and some of them are easier to recognize. So, in this method, recognition is not performed in a left-to-right scan, but follows a meaningful order which depends on the visual and lexical significance of the letters. Moreover, this order also follows an edge-toward-center movement, like in human vision [105]. Matching between symbolic and physical descriptions can be performed at the letter, feature and even sub-feature levels. As the system knows in advance what it is searching for, it can make use of high-level contextual knowledge to improve recognition, even at low-level stages. This system is an attempt to provide a general framework allowing efficient cooperation between low-level and high-level recognition processes.

5. Results

It is very difficult to give comparative results of the methods proposed so far. Most of the methods, and especially the early studies were tested on small databases created from the writing of only a few people. Moreover, all the databases used were different: no common test base was available, this simple fact making comparison quite perilous. This is the reason why we have decided not to include results in the previous sections which are only dedicated to the description of the techniques.

Common databases of cursive script samples have been made available in the recent past with the development of postal research in several countries. For instance, standard databases of checks or mail addresses were respectively created by the research department of the French Postal Service (SRTP) or by CEDAR at Buffalo for the US Postal Service. Furthermore, postal organisms have initiated and funded a large part of research on handwriting recognition. For instance, USPS recently funded a research project on cursive script recognition which involved five major private or academic laboratories. Although work is still under progress, many papers have already been published as well as preliminary results. In our opinion, these research programs constitute the "state of the art" in the field of off-line cursive script recognition and the available results provide a representative idea of the progress in this field. In these projects, recognition was performed on city or state names constituting lexicons of different sizes. Experiments were made on lexicons of three different sizes containing either an average of 10, 100 or 1000 words. Words contained in the databases were belonging to various stylistic categories (handprinted, discrete, cursive and mixed), about two thirds of them being cursive. For instance, the following recognition rates were reported by Kimura and coworkers in [59]:

— 93.5% (and 95.6% if considering the best two choices) for size 10 lexicons,

— 87.6% (93% if considering the best four choices) for size 100 lexicons,

— 80.9% (88.7% if considering the best six choices) for size 1000 lexicons.

On-line recognition constitutes another major field of CSR research. Unfortunately, fewer comparable results are available than in the case of off-line research. A large part of research remains unpublished (especially those concerning commercial products) as well as the corresponding results. Moreover, comparison between the different techniques is difficult because they have not been tested on a common database. The goals are also quite different: the lexicon is usually much larger (up to 20 000 words or even more) but the number of users is also much smaller (learning being often possible if not mandatory) and handwriting is also more constrained. So, recognition rates may vary considerably from one method to another (between 60% and 95%), depending on the size of the lexicon, the number of writers, the quality of the handwriting or whether training was or was not performed.

6. Conclusion

As a conclusion, we would like to make a double statement:

• it can now be said (as Kimura and coworkers write in [59]) that cursive script recognition has now become an achievable goal, provided that strong contextual constraints are available,

• contextual analysis constitutes the future of the field. Most of further improvements in CSR will now probably come from better lexical, syntactical and semantic analysis performed at the word, sentence and text levels.

This is already the direction that several major research teams have decided to follow [58][86][26] and there is no doubt that contextual analysis will be a field of intense research and achievements in the next few years.

References

1. G.H. Abbink, H.L. Teuling & L.R.B. Schomaker, *Description of on-line script using Hollerbach's generation model*, Pre-Procedings IWFHR III, page 217, Buffalo, May 1993.
2. A.M. Alimi & R. Plamondon, *Performance Analysis of Handwritten Strokes Generation Models*, Pre-Procedings IWFHR III, page 272, Buffalo, May 1993.
3. Bahl & Jelinek, *Decoding for channels with insertions, deletions & substitutions*, IEEE Trans. Inform. Theory vol.IT-21, pages 404-410, 1975.
4. E.J. Bellegarda, J. R. Bellegarda, D. Nahamoo, K.S. Nathan *A probabilistic framework for on-line handwriting recognition*, Pre-Procedings IWFHR III, page 225, Buffalo, May 1993.
5. S. Bercu, G. Lorette *On-line Handwritten Word Recognition: An Approach based on Hidden Markov Models*, Pre-Procedings IWFHR III, page 385, Buffalo, May 1993.
6. M. Berthod & S. Ahyan, *On Line Cursive Script Recognition: A Structural Approach with Learning*, 5th Int. Conf. on Pattern Recognition, pages 723-725, 1980.
7. M. Berthod, *On-line Analysis of Cursive Writing*, in Computer Analysis and perception. Vol. I. Visual Signals. Ch. IV, pages 55-80, CRC Press, 1982.
8. G. Boccignone, A. Chianese, L.P. Cordella and A. Marcelli, *Recovering Dynamic Information From Static Handwriting*, Pattern Recognition, Vol. 26, No. 3, page 409, 1993.
9. R. Bozinovic & S.N. Srihari, *A String Correction Algorithm for Cursive Script Recognition*, IEEE PAMI, Vol.4, No.6, pages 655-663, 1982.
10. R.M. Bozinovic & S.N.Srihari, *Off-Line Cursive Script Recognition*, IEEE Trans. on Pattern Analysis and Machine Intel., Vol. 11, No. 1, page 68, 1989.
11. M.K. Brown & S. Ganapathy, *Cursive Script Recognition*, Proc. of the Int. Conf. on Cybernetics & Society, 1980.
12. C.J.C. Burges, J.I. Be and C.R. Nohl, *Recognition of handwritten cursive postal words using neural networks*, USPS 5th Advanced Technology Conference, page A-117, Nov 1992.
13. J. Camillerapp, G. Lorette, G. Menier, H. Ouladj, J-C. Pettier, *On-line and off-line methods for cursive script recognition*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, Chapter 3, page 273, North-Holland, 1992.
14. R.G. Casey and G. Nagy, *Recursive segmentation and classification of composite patterns*, 6th Int. Conf. on Pattern Recognition, page 1023, 1982.
15. M.Y. Chen, A. Kundu, J. Zhou and S.N. Srihari, *Off-line handwritten word recognition using hidden Markov model*, USPS 5th Advanced Technology Conference, page 563, Nov 1992.
16. M.Y. Chen, A. Kundu, S.N. Srihari, *Unconstraint handwritten word recognition using continuous density variable duration hidden Markov model*, Proc IEEE Int Conference on Acoust, Speech, Signal Processing, April 1993.
17. M.Y. Chen, A. Kundu, *An Alternative to Variable Duration HMM in Handwritten Word Recognition*, Pre-Procedings IWFHR III, page 82, Buffalo, May 1993.
18. M. Cheriet, *Reading Cursive Script by Parts*, Pre-Procedings IWFHR III, page 403, Buffalo, May 1993.
19. D.S. Doermann and A. Rosenfeld, *Temporal Clues in Handwriting*, 11th IAPR Int. Conf. on Pattern Recognition, Vol. II, page 317, Aug. 1992.
20. D. Doermann and A. Rosenfeld, *The Interpretation and Recognition of Interfering Strokes*, Pre-Procedings IWFHR III, page 82, Buffalo, May 1993.
21. G. Dimauro, S. Impedovo and G. Pirlo, *From character to cursive script recognition: future trends in scientific research*, 11th IAPR Int. Conf. on Pattern Recognition, Vol. II, page 516, Aug. 1992.
22. C.E. Dunn and P.S.P. Wang, *Character segmenting techniques for handwritten text - a survey*, 11th IAPR Int. Conf. on Pattern Recognition, Vol. II, page 577, Aug. 1992.
23. L.D. Earnest, *Machine Recognition of Cursive Writing*, C.Cherry editor, Information Processing, page 462-466, Butterworth, London, 1962.
24. R.W. Ehrich & K.J. Koehler, *Experiments in the Contextual Recognition of Cursive Script*, IEEE Trans. on Computers Vol. 24, No. 2, page 182, 1975.
25. D.G. Elliman & I.T. Lancaster, *A Review of Segmentation and Contextual Analysis Techniques for Text Recognition*, Pattern Recognition, Vol. 23, No. 3/4, pages 337-346, 1990.
26. L.J. Evett, C.J. Wells, F.G. Keenan, T. Rose & R.J. Whitrow, *Using luiguistic information to aid handwriting recognition*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, Chapter 5, page 339, North-Holland, 1992.
27. R.F.H. Farag, *Word-Level Recognition Recognition of Cursive Script*, IEEE Trans. on Computers Vol. C-28, No. 2, pages 172-175, Feb. 1979.
28. C. Faure, *Traitement Automatique de l'Ecrit et du Document*, Congrès biennal de l'AFCET, Versailles, France, June 1993.
29. J.T. Favata and S.N. Srihari, *Recognition of general handwritten words using a hypothesis generation and reduction methodology*, USPS 5th Advanced Technology Conference, page 237, Nov 1992.
30. R. Fenrich, *Segmenting of Automatically located handwritten numeric strings*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, Chapter 1, page 47, North-Holland, 1992.
31. G.D. Forney Jr, *The Viterbi Algorithm*, Proc. IEEE Vol. 61, No. 3, page 268, March 1973.
32. T. Fujisaki, K. Nathan, W. Cho, H. Beigi, *On-line unconstrained handwriting recognition by a probabilistic method*, Pre-Procedings IWFHR III, page 235, Buffalo, May 1993.
33. A.M. Gillies, *Cursive Word Recognition using Hidden Markov Models*, USPS 5th Advanced Technology Conference, Nov 1992.
34. M. Gilloux and M. Leroux, *Recognition of cursive scripts amounts on postal cheques*, USPS 5th Advanced Technology Conference, page 545, Nov 1992.
35. M. Gilloux, J.M. Bertille, M. Leroux, *Recognition of Handwritten Words in a Limited Dynamic Vocabulary*, Pre-Procedings IWFHR III, page 417, Buffalo, 1993.
36. M. Gilloux, *Hidden Markov Models in Handwriting Recognition*, NATO/ASI Fundamentals in Handwriting Recognition (final text), Château de Bonas, France, June 1993.

37. N. Gorsky, *Off-line Recognition of Bad Quality Handwritten Words Using Prototypes*, NATO/ASI Fundamentals in Handwriting Recognition, page 155 (working book), Château de Bonas, France, June 1993.

38. V. Govindaraju, D. Wang and S.N. Srihari, *Using Temporal Information In Off-Line Word Recognition*, USPS 5th Advanced Technology Conference, Nov 1992.

39. I. Guyon, D. Henderson, P. Albrecht, Y. LeCun and J. Denker, *Writer independent and writer adaptative neural network for on-line character recognition*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, Chapter 5, page 493, North-Holland, 1992.

40. P.A.V. Hall & G.R. Dowling, *Approximate String Matching*, ACM Computings Surveys 12, 381-401, Dec. 1980.

41. L.D. Harmon, *Handwriting Reader Recognizes Whole Words*, Electronics pages 29-31, August 1962.

42. L.D. Harmon, *Automatic Recognition of Print and Script*, Procedings of the IEEE, Vol. 60, No. 10, pages 1165-1177, Oct. 72.

43. K.C. Hayes, *Reading Handwritten Words Using Hierarchical Relaxation*, Computer Graphics and Image Processing, Vol. 14, pages 344-364, 1980.

44. Y. He , M. Chen , A. Kundu *Handwritten Word Recognition Using HMM with Adaptative Length Viterbi Algorithm*, ICASSP-92, vol. 3, page 153, 1992.

45. C.A. Higgins and R. Whitrow, *On-Line Cursive Script Recognition*, Procedings Int. Conf. on Human-Computer Interaction - INTERACT'84, Elsevier, 1985.

46. C.A. Higgins and D.M. Ford, *On-line recognition of connected handwriting by segmentation and template matching*, 11th IAPR Int. Conf. on Pattern Recognition, Vol. II, page 200, Aug. 1992.

47. T.K. Ho, J.J. Hull and S.N. Srihari, *A Word Shape Analysis Approach to Recognition of Degraded Word Images*, Pattern Recognition Letters, No 13, page 821, 1992.

48. T.K. Ho, J.J. Hull and S.N. Srihari, *On Multiple Classifier Systems for Pattern Recognition*, 11th IAPR Int. Conf. on Pattern Recognition, Vol. II, page 84, Aug. 1992.

49. M. Holt, M Beglou & S. Datta, *Slant-independent letter segmentation for off-line cursive script recognition*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, page 41, North-Holland, 1992.

50. G.F. Houle, C. Radelat, S. Resnick and P. Bock, *Handwritten Word Recognition Using Collective Learning Systems Theory*, Pre-Procedings IWFHR III, page 92, Buffalo, May 1993.

51. Y.S. Huang, C.Y. Suen, *An optimal Method of Combining Multiple Classifiers for Unconstrained Handwritten Numeral Recognition*, Pre-Procedings IWFHR III, page 11, Buffalo, May 1993.

52. J.J. Hull & S.N. Srihari *Experiments in Text Recognition with Binary n-Gram and Viterbi Algorithm*, IEEE Trans. on Pattern Analysis and Machine Intel., Vol. 4, pages 520-530, Sept 1982.

53. J.J. Hull & S.N. Srihari, *A computational Approach to Visual Word Recognition: Hypothesis Generation and Testing*, Computer Vision and Pattern Recognition, pages 156-161, June 1986.

54. J. Hull and S. Srihari, *Recognition of Handwritten Words for Address Reading*, USPS Fourth Advanced Technology Conference, page 192, Nov 1991.

55. J. Hull, T.K. Ho, J. Favata, V. Govindaraju and S. Srihari, *Combination of segmentation-based and wholistic handwritten word recognition algorithms*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, Chapter 3, page 261, North-Holland, 1992.

56. J.J. Hull, *A Hidden Markov Model for Language Syntax in Text Recognition,* 11th IAPR Int. Conf. on Pattern Recognition, Vol. II, page 124, The Hague, The Netherlands, Aug./Sept. 1992.

57. J.J. Hull & Y. Li, *Word Recognition Result Interpretation using the Vector Space Model for Information Retrieval*, Proceeding of Symposium on Document Analysis and Information Retrieval, page 147, Las Vegas, USA, 1993.

58. J.J. Hull, *Language-Level Syntactic and Semantic Constraints Applied to Visual Word Recognition*, NATO/ASI Fundamentals in Handwriting Recognition, page 179 (working book), Château de Bonas, France, June 1993.

59. F. Kimura, M. Shridhar & N. Narasimhamurthi, *Lexicon Directed Segmentation-Recognition Procedure for Unconstrained Handwritten Words*, Pre-Procedings IWFHR III, page 122, Buffalo, May 1993.

60. A. Kundu, Yang He and P. Bahl, *Recognition of Handwriten Words: First and Second Order Hidden Markov Model Based Approach*, Pattern Recognition, Vol. 22, No. 3, page 283, 1989.

61. E. Lecolinet and J-V. Moreau, *Off-Line Recognition of Handwritten Cursive Script for the Automatic Reading of City Names on Real Mail*, 10th Int. Conf. on Pattern Recognition, page 674, Atlantic City, June 1990.

62. E. Lecolinet and J-P. Crettez, *A Grapheme-Based Segmentation Technique for Cursive Script Recognition*, Proceeding of the Int. Conf. on Document Analysis and Recognition, page 740, Saint Malo, France, Sept. 1991.

63. E. Lecolinet, *A New Model for Context-Driven Word Recognition*, Proceeding of Symposium on Document Analysis and Information Retrieval, page 135, Las Vegas, USA, April 1993.

64. E. Lecolinet, *Cursive Script Recognition by Backward Matching*, Proceeding of the Int. Conf. On Handwriting and Drawing (ICOHD), page 89, Paris, France, July 1993.

65. M. Leroux, J-C. Salome & J. Badard, *Recognition of cursive script words in a small lexicon*, Proceeding of the Int. Conf. on Document Analysis and Recognition, page 774, Saint Malo, France, Sept. 1991.

66. N. Lindgren, *Machine recognition of human language; Part III-Cursive script recognition*, IEEE Spectrum, pages 104-116, May 1965.

67. L. Li, T.K. Ho, J.J. Hull and S.N. Srihari, *A hypothesis testing approach to word recognition using dynamic feature selection*, 11th IAPR Int. Conf. on Pattern Recognition, Vol. II, page 586, Aug. 1992.

68. G. Lorette & Y. Lecourtier, *Is Recognition and Interpretation of Handwritten Text a Scene Analysis Problem?*, Pre-Proceedings IWFHR III, page 184, Buffalo, May 1993.

69. S. Madhvanath & V. Govindaraju, *Holisitic Lexicon Reduction*, Pre-Procedings IWFHR III, page 71, Buffalo, May 1993.

70. M. Maier, *Separating Characters in Scripted Documents*, 8th Int. Conf. on Pattern Recognition, Paris, pages 1056-58, 1986.

71. P. Mermelstein & M. Eden, *A System for Automatic Recognition of Handwritten Words*, IEEE Proceedings-Fall Joint Comp. Conf., 1964.

72. Miller, *Real time classification of handwritten script words*, Information Processing 71 - Proceedings of IFIP congress 71, Ljubljana, Yugoslavia, Aug. 1971.

73. P. Morasso, A. Pareto & S. Pagliano, *Neural models for handwriting recognition*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, page 423, North-Holland, 1992.

74. J-V. Moreau, B. Plessis, O. Bourgeois, J-L. Plagnaud, *A postal check reading system*, Proceeding of the Int. Conf. on Document Analysis and Recognition, page 758, Saint Malo, France, Sept. 1991.

75. R. Nag, K.H. Wong & F. Fallside, *Script Recognition Using Hidden Markov Models*, IEEE ICASSP, Tokyo, pages 2071-2074, 1986.

76. D.L. Neuhoff, *The Viterbi Algorithm As an Aid in Text Recognition*, IEEE Trans. on Inf. Theory, page 222, March 1975.

77. H. Nishida, T. Suzuki, S. Mori, *Thin Line Representation from Contour Representation of Handprinted Characters*, From Pixels to Features III, page 29, 1992.

78. H.Ouladj G.Lorette E.Petit J.Lemoine M.Gaudaire, *From Primitives to Letters : A Structural Method to Automatic Cursive Hanwritting Recognition*, The 6th Scandinavian Conference on Image Analysis (SCIA), Finland, pages 593-599, June 1989.

79. T. Paquet & Y. Lecourtier, *Handwriting recognition: application on bank cheques*, Proceeding of the Int. Conf. on Document Analysis and Recognition, page 749, Saint Malo, France, Sept. 1991.

80. M. Parizeau & R. Plamondon, *A Handwriting Model for Syntactic Recognition of Cursive Script*, Int. Conf. on Pattern Recognition, page 308, The Hague, 1992.

81. S. Peleg, *Ambiguity Reduction in Handwriting with Ambiguous Segmentation and Uncertain Interpretation*, Computer Graphics and Image Processing 10, pages 235-245, 1979.

82. R. Plamondon & P. Yergeau, *A System for the Analysis and Synthesis of Handwriting*, Procedings IWFHR I, page 167, Montreal, May 1990.

83. R. Plamondon, *A Model-Based Segmentation Framework for Computer Processing of Handwriting*, Int. Conf. on Pattern Recognition, page 303, The Hague, 1992.

84. B. Plessis, A. Sicsu, E. Menu, J-V. Moreau, *Isolated handwritten word recognition for contextual address reading*, USPS 5th Advanced Technology Conference, page 579, Nov 1992.

85. L.R. Rabiner and B.H. Juang *An introduction to Hidden Markov Models*, ASSP Mag. 3, pages 4-16, 1986.

86. R.K. Srihari, S.N. Charlotte, M. Baltus & J. Kud, *Use of Language Models in On-line Sentence / Phrase Recognition*, Pre-Procedings IWFHR III, page 284, Buffalo, May 1993.

87. K.M. Sayre, *Machine Recognition of Handwrittten Words : A Project Report*, Pattern Recognition, Vol. 5, pages 213-228, 1973.

88. A.W. Senior, *Off-line Handwriting Recognition: A Review and Experiments*, Technical Report TR 105, Cambridge University Engineering Department, Cambridge, England, Dec. 1992.

89. A.W. Senior, F. Fallside *An Off-line Cursive Script Recognition System Using Recurrent Error Propagation Networks*, Pre-Procedings IWFHR III, page 132, Buffalo, May 1993.

90. B.A. Sheil, *Median Split Trees: A Fast Lookup Technique for Frequently Occuring Keys*, Communications of the ACM, Vol. 221, page 947, Nov. 1978.

91. R. Shinghal & G.T. Toussaint, *Experiments in Text Recognition with the Modified Viterbi Algorithm*, IEEE PAMI Vol.1, No.2, page 184, April 1979.

92. R. Shinghal & G.T. Toussaint, *A Bottom-up and Top-down Approach to using Context in Text Recognition*, Int. J. Man-Machine Studies. Vol. 11, page 201, 1979.

93. M. Shridhar and A. Badreldin, *Recognition of Isolated and Simply Connected Handwritten Numerals*, Pattern Recognition, Vol.19, No.1, page 1, 1986.

94. J.C. Simon, K. Zerhouni, *Robust Description of a Line Image*, International Conference on Document Analysis and Recognition, page 3, 1991.

95. J.C. Simon and O. Baret, *Cursive Words Recognition*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, Chapter 3, page 241, North-Holland, 1992.

96. J.C. Simon, *Off-line cursive word recognition*, Proceedings of the IEEE, page 1150, July 1992.

97. J.C. Simon, *On the Robustness of Recognition of Degraded Line Images*, NATO/ASI Fundamentals in Handwriting Recognition, page 177 (working book), Château de Bonas, France, June 1993.

98. S.N. Srihari (Editor), *Computer Text Recognition and Error Correction (Tutorial)*, IEEE Computer Society, 1984.

99. S.N. Srihari & C.M. Baltus, *Combining Statistical and Syntactic Methods in Recognizing Handwritten Sentences,* Proceedings AAAI Symposium, page 121, San Jose, USA, 1992.

100. S.N. Srihari, *From Pixels to Paragraphs: the Use of Models in Text Recognition*, Proceeding of Symposium on Document Analysis and Information Retrieval, page 47, Las Vegas, USA, 1993.

101. C.Y. Suen, J. Guo & Z.C. Li, *Computer and human recognition of handprinted characters by parts*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, page 223, North-Holland, 1992.

102. T. Suzuki, S. Mori, *A thinning method based on cell structure*, Proceedings IWFHR I, Montreal, page 39, 1990.

103. C.C. Tappert, *Cursive Script Recognition by Elastic Matching*, IBM J. Res. Develop. 26, pages 765-771, Nov. 1982.

104. C.C. Tappert, C.Y. Suen and T. Wakahara, *The State of the Art in On-line Handwriting Recognition*, IEEE PAMI, Vol. 12, No. 8, page 787, August 1990.

105. I. Taylor and M. Taylor, *The Psychology of Reading*, Academic Press, 1983.

106. H.L. Teulings & L.R.B. Schomaker, *A Handwriting Recognition System Based on the Properties and Architectures of the Human Motor System*, Procedings IWFHR I, page 195, Montreal, May 1990.

107. H.L. Teulings & L.R.B. Schomaker, *Learning Prototypes in Cursive Handwriting*, From Pixels to Features III, S. Impedovo and J.C. Simon Editors, page 61, North-Holland, 1992.

108. H.L. Teulings, *Invariant Handwriting Features Useful in Cursive Script Recognition*, NATO/ASI Fundamentals in Handwriting Recognition (final text), Château de Bonas, France, June 1993.

109. R.A. Wagner & M.J. Fischer, *The String-to-String Correction Problem*, JACM vol. 21, no. 1, page 168, Jan 1974.

110. K.H. Wong & F. Fallside, *Dynamic Programming in the recognition of connected handwritten script*, 2nd Conf. on Artificial Intelligence Applications, pages 666-670, Dec. 1985.