A Survey of Methods and Strategies in Character Segmentation

Richard G. Casey and Eric Lecolinet, *Member, IEEE*

Abstract—Character segmentation has long been a critical area of the OCR process. The higher recognition rates for isolated characters vs. those obtained for words and connected character strings well illustrate this fact. A good part of recent progress in reading unconstrained printed and written text may be ascribed to more insightful handling of segmentation.

This paper provides a review of these advances. The aim is to provide an appreciation for the range of techniques that have been developed, rather than to simply list sources. Segmentation methods are listed under four main headings. What may be termed the "classical" approach consists of methods that partition the input image into subimages, which are then classified. The operation of attempting to decompose the image into classifiable units is called "dissection." The second class of methods avoids dissection, and segments the image either explicitly, by classification of prespecified windows, or implicitly by classification of subsets of spatial features collected from the image as a whole. The third strategy is a hybrid of the first two, employing dissection together with recombination rules to define potential segments, but using classification to select from the range of admissible segmentation possibilities offered by these subimages. Finally, holistic approaches that avoid segmentation by recognizing entire character strings as units are described.

Index Terms—Optical character recognition, character segmentation, survey, holistic recognition, Hidden Markov Models, graphemes, contextual methods, recognition-based segmentation.

1 Introduction

1.1 The Role of Segmentation in Recognition Processing

Character segmentation is an operation that seeks to decompose an image of a sequence of characters into subimages of individual symbols. It is one of the decision processes in a system for optical character recognition (OCR). Its decision, that a pattern isolated from the image is that of a character (or some other identifiable unit), can be right or wrong. It is wrong sufficiently often to make a major contribution to the error rate of the system.

In what may be called the "classical" approach to OCR, Fig. 1, segmentation is the initial step in a three-step procedure:

Given a starting point in a document image:

1) Find the next character image.
2) Extract distinguishing attributes of the character image.
3) Find the member of a given symbol set whose attributes best match those of the input, and output its identity.

This sequence is repeated until no additional character images are found.

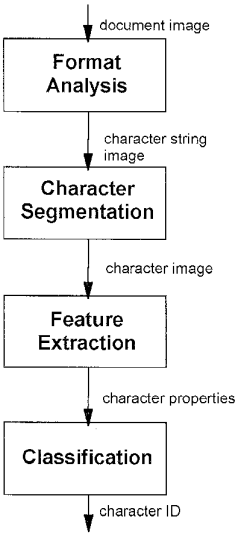


Fig. 1. Classical optical character recognition (OCR). The process of character recognition consists of a series of stages, with each stage passing its results on to the next in pipeline fashion. There is no feedback loop that would permit an earlier stage to make use of knowledge gained at a later point in the process.

An implementation of step 1, the segmentation step, requires answering a simply posed question: "What constitutes a character?" The many researchers and developers who have tried to provide an algorithmic answer to this question find themselves in a Catch-22 situation. A character is a pattern that resembles one of the symbols the system is designed to recognize. But to determine such a resem-

- *R.G. Casey is with the IBM Almaden Research Center, San Jose, CA 95120. E-mail: casey@ix.netcom.com.*
- *E. Lecolinet is with École Nationale Supérieure des Télécommunications (ENST), Paris, France.*

Manuscript received Mar. 18, 1995; revised Mar. 22, 1996. Recommended for acceptance by R. Kasturi.
For information on obtaining reprints of this article, please send e-mail to: transpami@computer.org, and reference IEEECS Log Number P96058.

blance the pattern must be segmented from the document image. Each stage depends on the other, and in complex cases it is paradoxical to seek a pattern that will match a member of the system's recognition alphabet of symbols without incorporating detailed knowledge of the structure of those symbols into the process.

Furthermore, the segmentation decision is not a local decision, independent of previous and subsequent decisions. Producing a good match to a library symbol is necessary, but not sufficient, for reliable recognition. That is, a poor match on a later pattern can cast doubt on the correctness of the current segmentation/recognition result. Even a series of satisfactory pattern matches can be judged incorrect if contextual requirements on the system output are not satisfied. For example, the letter sequence "cl" can often closely resemble a "d," but usually such a choice will not constitute a contextually valid result.

Thus, it is seen that the segmentation decision is interdependent with local decisions regarding shape similarity, and with global decisions regarding contextual acceptability. This sentence summarizes the refinement of character segmentation processes in the past 40 years or so. Initially, designers sought to perform segmentation as per the "classical" sequence listed above. As faster, more powerful electronic circuitry has encouraged the application of OCR to more complex documents, designers have realized that step 1 can not be divorced from the other facets of the recognition process.

In fact, researchers have been aware of the limitations of the classical approach for many years. Researchers in the 1960s and 1970s observed that segmentation caused more errors than shape distortions in reading unconstrained characters, whether hand- or machine-printed. The problem was often masked in experimental work by the use of databases of well-segmented patterns, or by scanning character strings printed with extra spacing. In commercial applications, stringent requirements for document preparation were imposed. By the beginning of the 1980s, workers were beginning to encourage renewed research interest [73] to permit extension of OCR to less constrained documents.

The problems of segmentation persist today. The well-known tests of commercial printed text OCR systems by the University of Nevada, Las Vegas [64], [65], consistently ascribe a high proportion of errors to segmentation. Even when perfect patterns, the bitmapped characters that are input to digital printers, were recognized, commercial systems averaged 0.5% spacing errors. This is essentially a segmentation error by a process that attempts to isolate a word subimage. The article [6] emphatically illustrates the woes of current machine-print recognition systems as segmentation difficulties increase (see Fig. 2). The degradation in performance of NIST tests of handwriting recognition on segmented [86] and unsegmented [88] images underscore the continuing need for refinement and fresh approaches in this area. On the positive side of the ledger, the study [29] shows the dramatic improvements that can be obtained when a thoroughgoing segmentation scheme replaces one of prosaic design.

Some authors previously have surveyed segmentation, often as part of a more comprehensive work, e.g., cursive recognition [36], [19], [20], [55], [58], [81], or document analysis [23], [29]. In the present paper, we present a survey whose focus is character segmentation, and which attempts to provide broad coverage of the topic.

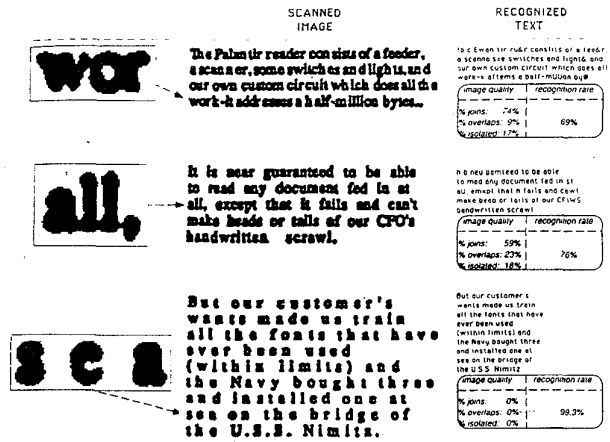


Fig. 2. Segmentation problems in machine print OCR. The figure illustrates performance of a 1991 prototype of a commercial reader as character spacing is varied. Proceeding from top to bottom, the images (at left) correspond to condensed, normal, and expanded spacing, respectively. Note that recognition output (at right) for the well-spaced images is quite accurate, but degrades badly when characters begin to merge. This system has continued to evolve and the illustration is not representative of performance of later releases. (Adapted from [6].)

1.2 Organization of Methods

A major problem in discussing segmentation is how to classify methods. Tappert et al. [81], for example, speaks of "external" vs. "internal" segmentation, depending on whether recognition is required in the process. Dunn and Wang [20] use "straight segmentation" and "segmentation-recognition" for a similar dichotomization.

A somewhat different point of view is proposed in this paper. The division according to use or nonuse of recognition in the process fails to make clear the fundamental distinctions among present-day approaches. For example, it is not uncommon in text recognition to use a spelling corrector as a post-processor. This stage may propose the substitution of two letters for a single letter output by the classifier. This is in effect a use of recognition to resegment the subimage involved. However, the process represents only a trivial advance on traditional methods that segment independent of recognition.

In this paper, the distinction between methods is based on how segmentation and classification interact in the overall process. In the example just cited, segmentation is done in two stages, one before and one after image classification. Basically an unacceptable recognition result is re-examined and modified by a (implied) resegmentation. This is a rather "loose" coupling of segmentation and classification.

A more profound interaction between the two aspects of recognition occurs when a classifier is invoked to select the segments from a set of possibilities. In this family of approaches, segmentation and classification are integrated. To

some observers it even appears that the classifier performs segmentation since, conceptually at least, it could select the desired segments by exhaustive evaluation of all possible sets of subimages of the input image.

After reviewing available literature, we have concluded that there are three "pure" strategies for segmentation, plus numerous hybrid approaches that are weighted combinations of these three. The elemental strategies are:

1) The classical approach, in which segments are identified based on "character-like" properties. This process of cutting up the image into meaningful components is given a special name, "dissection," in discussions below.
2) Recognition-based segmentation, in which the system searches the image for components that match classes in its alphabet.
3) Holistic methods, in which the system seeks to recognize words as a whole, thus, avoiding the need to segment into characters.

In strategy 1, the criterion for good segmentation is the agreement of general properties of the segments obtained with those expected for valid characters. Examples of such properties are height, width, separation from neighboring components, disposition along a baseline, etc. In method 2, the criterion is recognition confidence, perhaps including syntactic or semantic correctness of the overall result. Holistic methods (strategy 3) in essence revert to the classical approach with words as the alphabet to be read. The reader interested to obtain an early illustration of these basic techniques may glance ahead to Fig. 6 for examples of dissection processes, Fig. 13 for a recognition-based strategy, and Fig. 16 for a holistic approach.

Although examples of these basic strategies are offered below, much of the literature reviewed for this survey reports a blend of methods, using combinations of dissection, recognition searching, and word characteristics. Thus, although the paper necessarily has a discrete organization, the situation is perhaps better conceived as in Fig. 3. Here, the three fundamental strategies occupy orthogonal axes: Hybrid methods can be represented as weighted combinations of these lying at points in the intervening space. There is a continuous space of segmentation strategies rather than a discrete set of classes with well-defined boundaries. Of course, such a space exists only conceptually; it is not meaningful to assign precise weights to the elements of a particular combination.

Taking the fundamental strategies as a base, this paper is organized as indicated in Fig. 4. In Section 2, we discuss methods that are highly weighted towards strategy 1. These approaches perform segmentation based on general image features and then classify the resulting segments. Interaction with classification is restricted to reprocessing of ambiguous recognition results.

In Section 3, we present methods illustrative of recognition-based segmentation, strategy 2. These algorithms avoid early imposition of segmentation boundaries. As Fig. 4 shows, such methods fall into two subcategories. Windowing methods segment the image blindly at many boundaries chosen without regard to image features, and then try to choose an optimal segmentation by evaluating the classification of the subimages generated. Feature-based methods detect the physical location of image features, and seek to segment this representation into well-classified subsets. Thus, the former employs recognition to search for "hard" segmentation boundaries, the latter for "soft" (i.e., implicitly defined) boundaries.

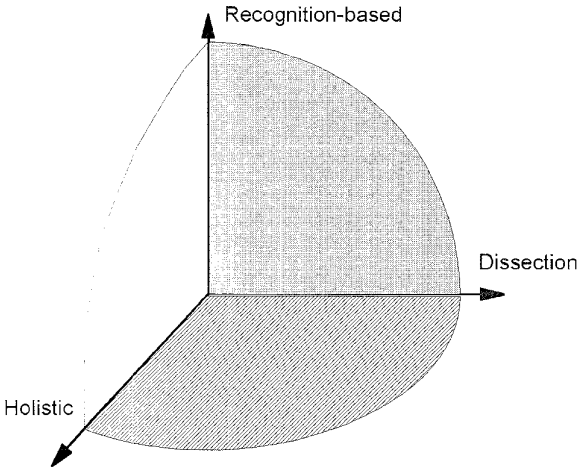


Fig. 3. The space of segmentation strategies. The fundamental segmentation strategies identified in the text are shown as occupying orthogonal axes. Many methods adopted in practice employ elements of more than one basic strategy. Although it is impossible to assign precise coordinates in this space, in concept such methods lie in the interior of the space bounded by the three planes shown.

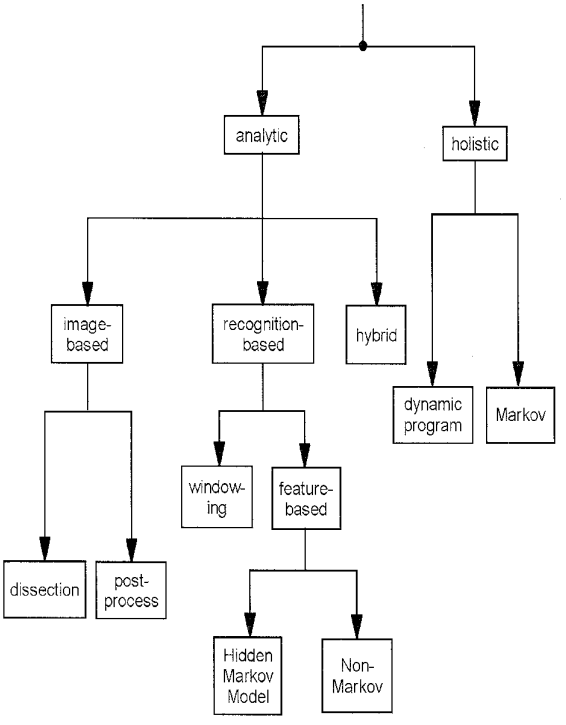


Fig. 4. Hierarchy of segmentation methods. The text follows the classification scheme shown.

Sections 2 and 3 describe contrasting strategies: one in which segmentation is based on image features, and a second in which classification is used to select from segmentation candidates generated without regard to image content. In Section 4, we discuss a hybrid strategy in which a preliminary segmentation is implemented, based on image features, but a later combination of the initial segments is performed and evaluated by classification in order to choose the best segmentation from the various hypotheses offered.

The techniques in Sections 2 to 4 are letter-based and, thus, are not limited to a specific lexicon. They can be applied to the recognition of any vocabulary. In Section 5, are presented holistic methods, which attempt to recognize a word as a whole. While avoiding the character segmentation problem, they are limited in application to a predefined lexicon. Markov models appear frequently in the literature, justifying further subclassification of holistic and recognition-based strategies, as indicated in Fig. 4. Section 6 offers some remarks and conclusions of the state of research in the segmentation area. Except when approaches of a sufficiently general nature are discussed, the discussion is limited to Western character sets as well as to "off-line" character recognition, that is to segmentation of character data obtained by optical scanning rather than at inception ("on-line") using tablets, light pens, etc. Nevertheless, it is important to realize that much of the thinking behind advanced work is influenced by related efforts in speech and on-line recognition, and in the study of human reading processes.

2 Dissection Techniques for Segmentation

Methods discussed in this section are based on what will be termed *"dissection"* of an image. By *dissection* is meant the decomposition of the image into a sequence of subimages using general features (as, for example, in Fig. 5). This is opposed to later methods that divide the image into subimages independent of content. Dissection is an intelligent process in that an analysis of the image is carried out; however, classification into symbols is not involved at this point.

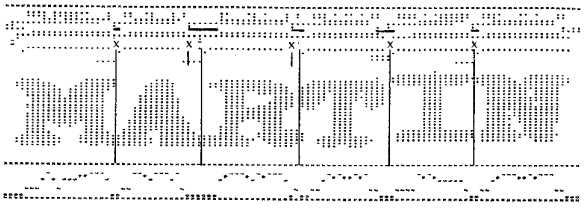


Fig. 5. The dissection method of Hoffman and McCullough. An evaluation function based on a running count of horizontal black-white and white-black transitions is plotted below the image. The horizontal black bars above the image indicate the activation region for the function. Vertical lines indicate human estimates of optimal segmentation (from [43]) columns, while Xs indicate columns chosen by the algorithm.

In literature describing systems where segmentation and classification do not interact, dissection is the entire segmentation process: The two terms are equivalent. However, in many current studies, as we shall see, segmentation is a complex process, and there is a need for a term such as "dissection" to distinguish the image-cutting subprocess from the overall segmentation, which may use contextual knowledge and/or character shape description.

2.1 Dissection Directly into Characters

In the late 1950s and early 1960s, during the earliest attempts to automate character recognition, research was focused on the identification of isolated images. Workers mainly sought methods to characterize and classify character shapes. In some cases, individual characters were mapped onto grids and pixel coordinates entered on punched cards [40], [49] in order to conduct controlled development and testing. As CRT scanners and magnetic storage became available, well-separated characters were scanned, segmented by dissection based on whitespace measurement, and stored on tape. When experimental devices were built to read actual documents they dealt with constrained printing or writing that facilitated segmentation.

For example, bank check fonts were designed with strong leading edge features that indicated when a character was properly registered in the rectangular array from which it was recognized [24]. Handprinted characters were printed in boxes that were invisible to the scanner, or else the writer was constrained in ways that aided both segmentation and recognition. A very thorough survey of status in 1961 [79] gives only implicit acknowledgment of the existence of the segmentation problem. Segmentation is not shown at all in the master diagram constructed to accompany discussion of recognition stages. In the several pages devoted to preprocessing (mainly thresholding), the function is indicated only peripherally as part of the operation of registering a character image.

The twin facts that early OCR development dealt with constrained inputs, while research was mainly concerned with representation and classification of individual symbols, explains why segmentation is so rarely mentioned in pre-70s literature. It was considered a secondary issue.

2.1.1 White Space and Pitch

In machine printing, vertical whitespace often serves to separate successive characters. This property can be extended to handprint by providing separated boxes in which to print individual symbols. In applications such as billing, where document layout is specifically designed for OCR, additional spacing is built into the fonts used. The notion of detecting the vertical white space between successive characters has naturally been an important concept in dissecting images of machine print or handprint.

In many machine print applications involving limited font sets, each character occupies a block of fixed width. The pitch, or number of characters per unit of horizontal distance, provides a basis for estimating segmentation points. The sequence of segmentation points obtained for a given line of print should be approximately equally spaced at the distance corresponding to the pitch. This provides a global basis for segmentation, since separation points are not independent.

Applying this rule permits correct segmentation in cases where several characters along the line are merged or broken. If most segmentations can be obtained by finding columns of white, the regular grid of intercharacter boundaries can be estimated. Segmentation points not lying near these boundaries can be rejected as probably due to broken characters. Segmentation points missed due to merged characters can be estimated as well, and a local analysis conducted in order to decide where best to split the composite pattern.

One well-documented early commercial machine that dealt with a relatively unconstrained environment was the reader IBM installed at the U.S. Social Security Administration in 1965 [38]. This device read alphanumeric data typed by employers on forms submitted quarterly to the SSA. There was no way for SSA to impose constraints on the printing process. Typewriters might be of any age or condition, ribbons in any state of wear, and the font style might be one or more of approximately 200.

In the SSA, reader segmentation was accomplished in two scans of a print line by a flying-spot scanner. On the initial scan, from left to right, the character pitch distance D was estimated by analog circuitry. On the return scan, right to left, the actual segmentation decisions were made using parameter D. The principal rule applied was that a double white column triggered a segmentation boundary. If none was found within distance D, then segmentation was forced.

Hoffman and McCullough [43] generalized this process and gave it a more formal framework (see Fig. 5). In their formulation, the segmentation stage consisted of three steps:

1) Detection of the start of a character.
2) A decision to begin testing for the end of a character (called sectioning).
3) Detection of end-of-character.

Sectioning, step 2, was the critical step. It was based on a weighted analysis of horizontal black runs completed, versus runs still incomplete as the print line was traversed column-by-column. An estimate of character pitch was a parameter of the process, although in experiments it was specified for 12-character per inch typewriting. Once the sectioning algorithm indicated a region of permissible segmentation, rules were invoked to segment based on either an increase in bit density (start of a new character) or else on special features designed to detect end-of-character. The authors experimented with 80,000 characters in 10- and 12-pitch serif fonts containing 22% touching characters. Segmentation was correct to within one pixel about 97% of the time. The authors noted that the technique was heavily dependent on the quality of the input images, and tended to fail on both very heavy or very light printing.

2.1.2 Projection Analysis

The vertical projection (also called the "vertical histogram") of a print line, Fig. 6a, consists of a simple running count of the black pixels in each column. It can serve for detection of white space between successive letters. Moreover, it can indicate locations of vertical strokes in machine print, or regions of multiple lines in handprint. Thus, analysis of the projection of a line of print has been used as a basis for segmentation of noncursive writing. For example, in [66], in segmenting Kanji handprinted addresses, columns where the projection fell below a predefined threshold were candidates for splitting the image.

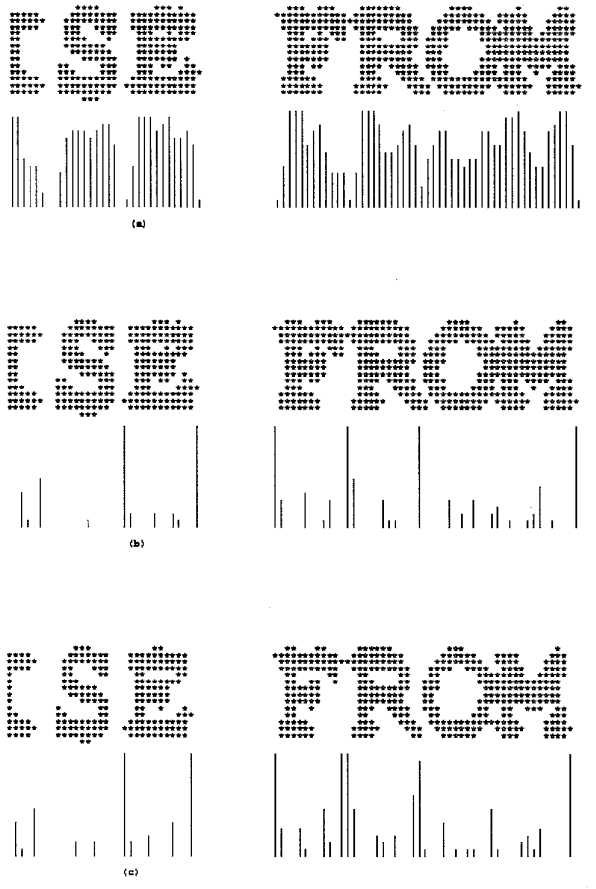


Fig. 6. Dissection based on projection: (a) Vertical projection of an image. It is an easy matter to detect white columns between characters, and regions of light contact. The function fails, however, to make clear the O-M separation. (b) Differencing measure for column splitting. The function from [1] is based on a second difference of the projection. This gives a clear peak at most separation columns, but may still fail for the O-M case. (c) Differencing measure after column ANDing. The image transformed by an AND of adjacent columns, with the difference function of (b) applied to the transformed image. The AND operation tends to increase separation, leading to a better defined peak at the O-M boundary.

When printed characters touch, or overlap horizontally, the projection often contains a minimum at the proper segmentation column. In [1], the projection was first obtained, then the ratio of second derivative of this curve to its height was used as a criterion for choosing separating columns (see Fig. 6b). This ratio tends to peak at minima of the projection, and avoids the problem of splitting at points along thin horizontal lines.

A peak-to-valley function was designed to improve on this method in [59]. A minimum of the projection is located and the projection value noted. The sum of the differences

between this minimum value and the peaks on each side is calculated. The ratio of the sum to the minimum value itself (plus 1, presumably to avoid division by zero) is the discriminator used to select segmentation boundaries. This ratio exhibits a preference for low valley with high peaks on both sides.

A prefiltering was implemented in [83] in order to intensify the projection function. The filter ANDed adjacent columns prior to projection as in Fig. 6c. This has the effect of producing a deeper valley at columns where only portions of the vertical edges of two adjacent characters are merged.

A different kind of prefiltering was used in [57] to sharpen discrimination in the vicinity of holes and cavities of a composite pattern. In addition to the projection itself, the difference between upper and lower profiles of the pattern was used in a formula analogous to that of [1]. Here, the "upper profile" is a function giving the maximum y-value of the black pixels for each column in the pattern array. The lower profile is defined similarly on the minimum y-value in each column.

A vertical projection is less satisfactory for the slanted characters commonly occurring in handprint. In one study [28], projections were performed at two-degree increments between −16 and +16 degrees from the vertical. Vertical strokes and steeply angled strokes such as occur in a letter A were detected as large values of the derivative of a projection. Cuts were implemented along the projection angle. Rules were implemented to screen out cuts that traversed multiple lines, and also to rejoin small floating regions, such as the left portion of a T crossbar, that might be created by the cutting algorithm. A similar technique is employed in [89].

2.1.3 Connected Component Processing

Projection methods are primarily useful for good quality machine printing, where adjacent characters can ordinarily be separated at columns. A one-dimensional analysis is feasible in such a case.

However, the methods described above are not generally adequate for segmentation of proportional fonts or handprinted characters. Thus, pitch-based methods can not be applied when the width of the characters is variable. Likewise, projection analysis has limited success when characters are slanted, or when inter-character connections and character strokes have similar thickness.

Segmentation of handprint or kerned machine printing calls for a two-dimensional analysis, for even nontouching characters may not be separable along a single straight line. A common approach (see Fig. 7) is based on determining connected black regions ("connected components," or "blobs"). Further processing may then be necessary to combine or split these components into character images.

There are two types of followup processing. One is based on the "bounding box," i.e., the location and dimensions of each connected component. The other is based on detailed analysis of the images of the connected components.

2.1.3.1 Bounding Box Analysis

The distribution of bounding boxes tells a great deal about the proper segmentation of an image consisting of noncursive characters. By testing their adjacency relationships to perform merging, or their size and aspect ratios to trigger splitting mechanisms, much of the segmentation task can be accurately performed at a low cost in computation.

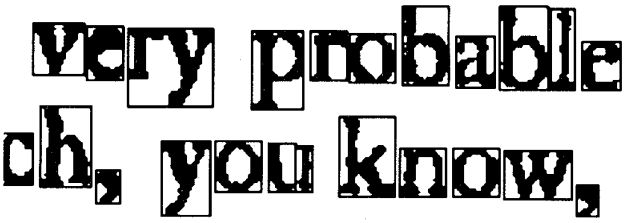


Fig. 7. Connected components. This example illustrates characters that consist of two components (e.g., the "u" in "you"), as well as components consisting of more than one character (e.g., the "ry" in "very"). The bounding box of each component is also shown. The latter is often used as a basis for dissection methods, as discussed in the text.

This approach has been applied, for example, in segmenting handwritten postcodes [14] using knowledge of the number of symbols in the code: six for the Canadian codes used in experiments. Connected components were joined or split according to rules based on height and width of their bounding boxes. The rather simple approach correctly classified 93% of 300 test codes, with only 2.7% incorrect segmentation and 4.3% rejection.

Connected components have also served to provide a basis for the segmentation of scanned handwriting into words [74]. Here, it is assumed that words do not touch, but may be fragmented. Thus, the problem is to group fragments (connected components) into word images. Eight different distance measures between components were investigated. The best methods were based on the size of the white run lengths between successive components, with a reduction factor applied to estimated distance if the components had significant horizontal overlap. 90% correct word segmentation was achieved on 1,453 postal address images.

An experimental comparison of character segmentation by projection analysis vs. segmentation by connected components is reported in [87]. Both segmenters were tested on a large data base (272,870 handprinted digits) using the same follow-on classifier. Characters separated by connected component segmentation resulted in 97.5% recognition accuracy, while projection analysis (along a line of variable slope) yielded almost twice the errors at 95.3% accuracy. Connected component processing was also carried out four time faster than projection analysis, a somewhat surprising result.

2.1.3.2 Splitting of Connected Components

Analysis of projections or bounding boxes offers an efficient way to segment nontouching characters in hand- or machine-printed data. However, more detailed processing is necessary in order to separate joined characters reliably. The intersection of two characters can give rise to special

image features. Consequently dissection methods have been developed to detect these features and to use them in splitting a character string image into subimages. Such methods often work as a follow-on to bounding box analysis. Only image components failing certain dimensional tests are subjected to detailed examination.

Another concern is that separation along a straight-line path can be inaccurate when the writing is slanted, or when characters are overlapping. Accurate segmentation calls for an analysis of the shape of the pattern to be split, together with the determination of an appropriate segmentation path (see Fig. 8).

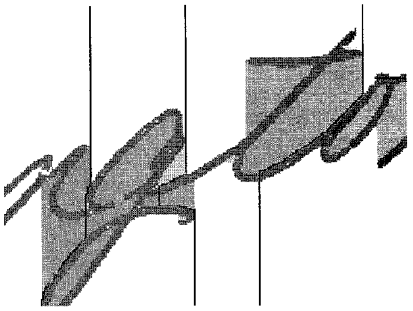


Fig. 8. Dissection based on search and deflect. The initial path of the cut trajectory is along a column corresponding to a minimum in the upper profile of the image. When black is encountered the path is modified recursively, seeking better positions at which to enforce a cut. In this way, multiple cuts can be made at positions which are in the shadow region of the image.

Depending upon the application, certain a priori knowledge may be used in the splitting process. For example, an algorithm may assume that some but not all input characters can be connected. In an application such as form data, the characters may be known to be digits or capital letters, placing a constraint on dimensional variations. Another important case commercially is handwritten zip codes, where connected strings of more than two or three characters are rarely encountered, and the total number of symbols is known.

These different points have been addressed by several authors. One of the earliest studies to use contour analysis for the detection of likely segmentation points was reported in [69]. The algorithm, designed to segment digits, uses local vertical minima encountered in following the bottom contour as "landmark points." Successive minima detected in the same connected component are assumed to belong to different characters, which are to be separated. Consequently, contour following is performed from the leftmost minimum counter-clockwise until a turning point is found. This point is presumed to lie in the region of intersection of the two characters and a cut is performed vertically. An algorithm that not only detects likely segmentation points, but also computes an appropriate segmentation path, as in Fig. 8, was proposed in [76] for segmenting digit strings. The operation comprises two steps. First, a vertical scan is performed in the middle of an image assumed to contain two possibly connected characters. If the number of black to white transitions is 0 or 2, then the digits are either not connected or else simply connected, respectively, and can therefore be separated easily by means of a vertical cut. If the number of transitions found during this scan exceeds 2, the writing is probably slanted and a special algorithm using a "Hit and Deflect Strategy" is called. This algorithm is able to compute a curved segmentation path by iteratively moving a scanning point. This scanning point starts from the maximum peak in the bottom profile of the lower half of the image. It then moves upwards by means of simple rules which seek to avoid cutting the characters until further movement is impossible. In most cases, only one cut is necessary to separate slanted characters that are simply connected.

This scheme was refined in a later technique [51], [52], which determines not only "how" to segment characters but also "when" to segment them. Detecting which characters have to be segmented is a difficult task that has not always been addressed. One approach consists in using recognition as a validation of the segmentation phase and resegmenting in case of failure. Such a strategy will be addressed in Section 3. A different approach, based on the concept of prerecognition, is proposed in [51].

The basic idea of the technique is to follow connected component analysis with a simple recognition logic whose role is not to label characters but rather to detect which components are likely to be single, connected or broken characters. Splitting of an image classified as connected is then accomplished by finding characteristic landmarks of the image that are likely to be segmentation points, rejecting those that appear to be situated within a character, and implementing a suitable cutting path.

The method employs an extension of the Hit and Deflect scheme proposed in [76]. First, the valleys of the upper and lower profiles of the component are detected. Then, several possible segmentation paths are generated. Each path must start from an upper or lower valley. Several heuristic criteria are considered for choosing the "best path" (the path must be "central" enough, paths linking an upper and a lower valley are preferred, etc.).

The complete algorithm works as a closed loop system, all segmentation being proposed and then confirmed or discarded by the prerecognition module: Segmentation can only take place on components identified as connected characters by prerecognition, and segmentations producing broken characters are discarded. The system is able to segment n-tuplets of connected characters which can be multiply connected or even merged. It was first applied on zip code segmentation for the French Postal Service.

In [85], an algorithm was constructed based on a categorization of the vertexes of stroke elements at contact points of touching numerals. Segmentation consists in detecting the most likely contact point among the various vertexes proposed by analysis of the image, and performing a cut similar in concept to that illustrated in Fig. 8.

Methods for defining splitting paths have been examined in a number of other studies as well. The algorithm of [17] performs background analysis to extract the face-up and face-down valleys, strokes and loop regions of component images. A "marriage score matrix" is then used to decide which pair of valleys is the most appropriate. The

separating path is deduced by combining three lines respectively segmenting the upper valley, the stroke and the lower valley.

A distance transform is applied to the input image in [31] in order to compute the splitting path. The objective is to find a path that stays as far from character strokes as possible without excessive curvature. This is achieved by employing the distance transform as a cost function, and using complementary heuristics to seek a minimal-cost path.

A shortest-path method investigated in [84] produces an "optimum" segmentation path using dynamic programming. The path is computed iteratively by considering successive rows in the image. A one-dimensional cost array contains the accumulated cost of a path emanating from a predetermined starting point at the top of the image to each column of the current row. The costs to reach the following row are then calculated by considering all vertical and diagonal moves that can be performed from one point of the current row to a point of the following row (a specific cost being associated to each type of move). Several tries can be made from different starting points. The selection of the best solution is based on classification confidence (which is obtained using a neural network). Redundant shortest-path calculations are avoided in order to improve segmentation speed.

2.1.3.3 Landmarks

In recognition of cursive writing, it is common to analyze the image of a character string in order to define lower, middle and upper zones. This permits the ready detection of ascenders and descenders, features that can serve as "landmarks" for segmentation of the image. This technique was applied to on-line recognition in pioneering work by Frischkopf and Harmon [36]. Using an estimate of character width, they dissected the image into patterns centered about the landmarks, and divided remaining image components on width alone. This scheme does not succeed with letters such as "u," "n," "m," which do not contain landmarks. However, the basic method for detecting ascenders and descenders has been adopted by many other researchers in later years.

2.2 Dissection with Contextual Postprocessing: Graphemes

The segmentation obtained by dissection can later be subjected to evaluation based on linguistic context, as shown in [7]. Here, a Markov model is postulated to represent splitting and merging as well as misclassification in a recognition process. The system seeks to correct such errors by minimizing an edit distance between recognition output and words in a given lexicon. Thus, it does not directly evaluate alternative segmentation hypotheses, it merely tries to correct poorly made ones. The approach is influenced by earlier developments in speech recognition. A non-Markovian system reported in [12] uses a spell-checker to correct repeatedly made merge and split errors in a complete text, rather than in single words as above.

An alternative approach still based on dissection is to divide the input image into subimages that are not necessarily individual characters. The dissection is performed at stable image features that may occur within or between characters, as for example, a sharp downward indentation can occur in the center of an "M" or at the connection of two touching characters. The preliminary shapes, called "graphemes" or "pseudo-characters" (see Fig. 9), are intended to fall into readily identifiable classes. A contextual mapping function from grapheme classes to symbols can then complete the recognition process. In doing so, the mapping function may combine or split grapheme classes, i.e., implement a many-to-one or one-to-many mapping. This amounts to an (implicit) resegmentation of the input. The dissection step of this process is sometimes called "presegmentation" or, when the intent is to leave no composite characters, "over-segmentation."

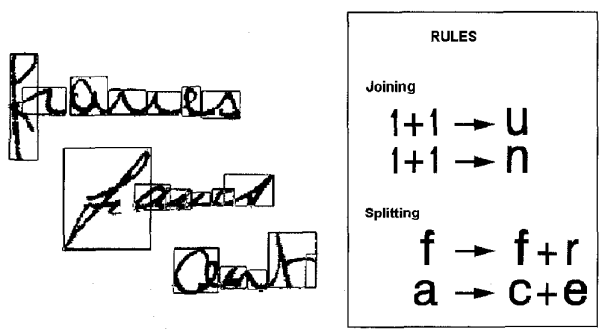


Fig. 9. Graphemes. The bounding boxes indicate subimages isolated by a cutting algorithm. The algorithm typically cuts "u" or "n" into two pieces as shown, but this is anticipated by the contextual recombination rules at right. Other rules are shown for cases where the cutting algorithm failed to split two characters. These rules are applied in many combinations seeking to transform the grapheme classes obtained during recognition into a valid character string.

The first reported use of this concept was probably in [72], a report on a system for off-line cursive script recognition. In this work, a dissection into graphemes was first performed based on the detection of characteristic areas of the image. The classes recognized by the classifier did not correspond to letters, but to specific shapes that could be reliably segmented (typically combinations of letters, but also portions of letters). Consequently, only 17 nonexclusive classes were considered.

As in [72], the grapheme concept has been applied mainly to cursive script by later researchers. Techniques for dissecting cursive script are based on heuristic rules derived from visual observation. There is no "magic" rule and it is not feasible to segment all handwritten words 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 (the graphemes). The relationship between characters and graphemes must remain simple enough to allow definition of an efficient post-processing stage. In practice, this means that a single character decomposes into at most two graphemes, and conversely, a single grapheme represents at most a two- or three-character sequence.

The line segments that form connections between characters in cursive script are known as "ligatures." Thus,

some dissection techniques for script seek "lower ligatures," connections near the baseline that link most lowercase characters. A simple way to locate ligatures is to detect the minima of the upper outline of words. Unfortunately, this method leaves several problems unresolved:

- Letters "o," "b," "v," and "w" are usually followed by "upper" ligatures.
- Letters "u" and "w" contain "intraletter ligatures," i.e., a subpart of these letters cannot be differentiated from a ligature in the absence of context.
- Artifacts sometimes cause erroneous segmentation.

In typical systems, these problems are treated at a later contextual stage that jointly treats both segmentation and recognition errors. Such processing is included in the system since cursive writing is often ambiguous without the help of lexical context. However, the quality of segmentation still remains very much dependent on the effectiveness of the dissection scheme that produces the graphemes.

Dissection techniques based on the principle of detecting ligatures were developed in [22], [61], and [53]. The last study was based on a dual approach:

- The detection of possible presegmentation zones.
- The use of a "prerecognition" algorithm, whose aim was not to recognize characters, but to evaluate whether a subimage defined by the presegmenter was likely to constitute a valid character.

Presegmentation zones were detected by analyzing the upper and lower profiles and open concavities of the words. Tentative segmentation paths were defined in order to separate words into isolated graphemes. These paths were chosen to respect several heuristic rules expressing continuity and connectivity constraints. However, these presegmentations were only validated if they were consistent with the decisions of the prerecognition algorithm. An important property of this method was independence from character slant, so that no special preprocessing was required.

A similar presegmenter was presented in [42]. In this case, analysis of the upper contour, and a set of rules based on contour direction, closure detection, and zone location were used. Upper contour analysis was also used in [47] for a presegmentation algorithm that served as part of the second stage of a hybrid recognition system. The first stage of this system also implemented a form of the hit and deflect strategy previously mentioned.

A technique for segmenting handwritten strings of variable length, was described in [27]. It employs upper and lower contour analysis and a splitting technique based on the hit and deflect strategy.

Segmentation can also be based on the detection of minima of the lower contour as in [8]. In this study, presegmentation points were chosen in the neighborhood of these minima and emergency segmentation performed between points that were highly separated. The method requires handwriting to be previously deslanted in order to ensure proper separation.

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

3 Recognition-Based Segmentation

Methods considered here also segment words into individual units (which are usually letters). However, the principle of operation is quite different. In principle, no feature-based dissection algorithm is employed. Rather, the image is divided systematically into many overlapping pieces without regard to content. These are classified as part of an attempt to find a coherent segmentation/recognition result. Systems using such a principle perform "recognition-based" segmentation: Letter segmentation is a by-product of letter recognition, which may itself be driven by contextual analysis. The main interest of this category of methods is that they bypass the segmentation problem: No complex "dissection" algorithm has to be built and recognition errors are basically due to failures in classification. The approach has also been called "segmentation-free" recognition. The point of view of this paper is that recognition necessarily involves segmentation, explicit or implicit though it be. Thus, the possibly misleading connotations of "segmentation-free" will be avoided in our own terminology.

Conceptually, these methods are derived from a scheme in [48] and [11] for the recognition of machine-printed words. The basic principle is to use a mobile window of variable width to provide sequences of tentative segmentations which are confirmed (or not) by character recognition. Multiple sequences are obtained from the input image by varying the window placement and size. Each sequence is assessed as a whole based on recognition results.

In recognition-based techniques, recognition can be performed by following either a serial or a parallel optimization scheme. In the first case, e.g., [11], recognition is done iteratively in a left-to-right scan of words, searching for a "satisfactory" recognition result. The parallel method [48] proceeds in a more global way. It generates a lattice of all (or many) possible feature-to-letter combinations. The final decision is found by choosing an optimal path through the lattice.

The windowing process can operate directly on the image pixels, or it can be applied in the form of weightings or groupings of positional feature measurements made on the images. Methods employing the former approach are presented in Section 3.1, while the latter class of methods is explored in Section 3.2.

Word level knowledge can be introduced during the recognition process in the form of statistics, or as a lexicon of possible words, or by a combination of these tools. Statistical representation, which has become popular with the use of Hidden Markov Models (HMMs), is discussed in Section 3.2.1.

3.1 Methods that Search the Image

Recognition-based segmentation consists of the following two steps:

1) Generation of segmentation hypotheses (windowing step).
2) Choice of the best hypothesis (verification step).

How these two operations are carried out distinguishes the different systems.

As easy to state as these principles are, they were a long time in developing. Probably the earliest theoretical and experimental application of the concept is reported by Kovalevsky [48]. The task was recognition of typewritten Cyrillic characters of poor quality. Although character spacing was fixed, Kovalevski's model assumed that the exact value of pitch and the location of the origin for the print line were known only approximately. He developed a solution under the assumption that segmentation occurred along columns. Correlation with prototype character images was used as a method of classification.

Kovalevsky's model (Fig. 10) assumes that the probability of observing a given version of a prototype character is a spherically symmetric function of the difference between the two images. Then the optimal objective function for segmentation is the sum of the squared distances between segmented images and matching prototypes. The set of segmented images that minimizes this sum is the optimal segmentation. He showed that the problem of finding this solution can be formulated as one of determining the path of maximum length in a graph, and that this path can be found by dynamic programming. This process was implemented in hardware to produce a working OCR system.

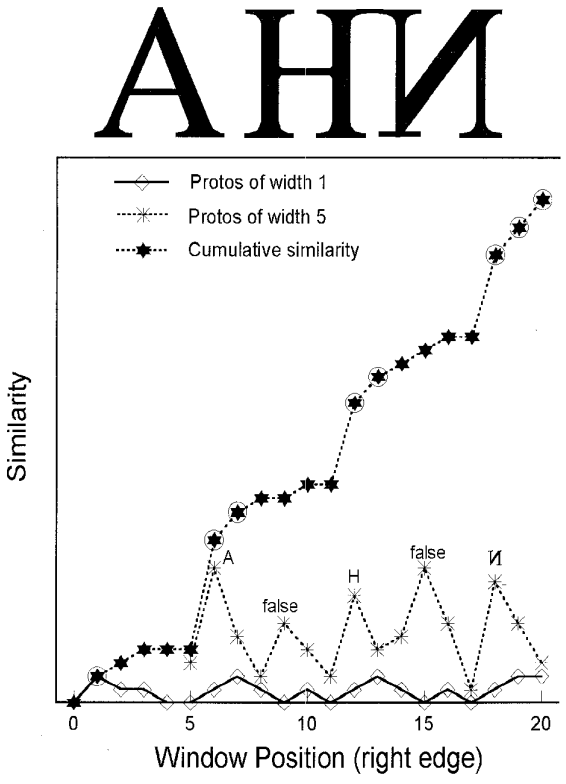


Fig. 10. Kovalevsky's method. The lower two curves give the similarities obtained by comparing stored prototypes against a window whose right edge is placed on the input image (shown at top) at the positions indicated by the abcissa. For window size 1 the only prototype is a blank column. For windows of size 5 there are multiple character prototypes, and the highest similarity among these is plotted. The cumulative plot gives the maximum total similarity obtainable from a sequence of windows up to the plotted point. Several good matches are obtainable by windowing parts of two characters. However, such "false" matches are not part of the optimal sequence of windows, indicated by the circled points of the upper graph.

Kovalevsky's work appears to have been neglected for some time. A number of years later, [11] reported a recursive splitting algorithm for machine-printed characters. This algorithm, also based on prototype matching, systematically tests all combinations of admissible separation boundaries until it either exhausts the set of cutpoints, or else finds an acceptable segmentation (see Fig. 11). An acceptable segmentation is one in which every segmented pattern matches a library prototype within a prespecified distance tolerance.

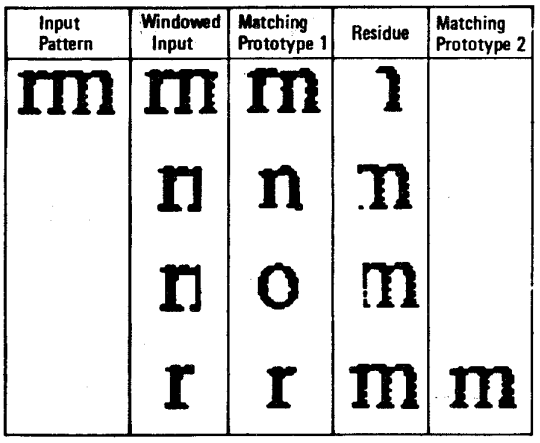


Fig. 11. Recursive segmentation. The example shows the results of applying windows of decreasing width to the left side of an input image. When the subimage in the window is recognized (in this case by matching a prototype character stored in the system's memory), then the procedure is recursively applied to the residue image. Recognition (and segmentation) is accomplished if a complete series of matching windows is found. In the top three rows, no match is obtained for the residue image, but successful segmentation is finally obtained as shown at the bottom.

A technique combining dynamic programming and neural net recognition was proposed in [10]. This technique, called "Shortest Path Segmentation," selects the optimal consistent combination of cuts from a predefined set of windows. Given this set of candidate cuts, all possible "legal" segments 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 "distance" obtained by the neural net recognizer. The shortest path through the graph thus corresponds to the best recognition and segmentation of the word.

The method of "selective attention" [30] takes neural networks even further in the handling of segmentation problems. In this approach, Fig. 12, a neural net seeks recognizable patterns in an image input, but is inhibited automatically after recognition in order to ignore the region of the recognized character and search for new character images in neighboring regions.

Fig. 12. Selective attention. (a) An input pattern. (b) The recognizer gradually reinforces pixels that corresponds to objects in its template library, and inhibits those that do not, yielding a partial recognition. (c) After a delay, attention is switched to unmatched regions, and another match to the library is found (after Fukishima).

3.2 Methods that Segment a Feature Representation of the Image

3.2.1 Hidden Markov Models

A Hidden Markov Model (often abbreviated HMM) models variations in printing or cursive writing as an underlying probabilistic structure which is not directly observable. This structure consists of a set of states plus transition probabilities between states. In addition, the observations that the system makes on an image are represented as random variables whose distribution depends on the state. These observations constitute a sequential feature representation of the input image. The survey [34] provides an introduction to its use in recognition applications.

For the purpose of this survey, three levels of underlying Markov model are distinguished, each calling for a different type of feature representation:

1) The Markov model represents letter-to-letter variations of the language. Typically such a model is based on bigram frequencies (first order model) or trigram frequencies (second order model). The features are gathered on individual characters or graphemes, and segmentation must be done in advance by dissection. Such systems are included in Section 2 above.
2) The Markov model represents state-to-state transitions within a character. These transitions provide a sequence of observations on the character. Features are typically measured in the left-to-right direction. This facilitates the representation of a word as a concatenation of character models. In such a system, segmentation is (implicitly) done in the course of matching the model against a given sequence of feature values gathered from a word image. That is, it decides where one character model leaves off and the next one begins, in the series of features analyzed. Examples of this approach are given in this section.
3) The Markov model represents the state-to-state variations within a specific word belonging to a lexicon of admissible word candidates. This is a holistic model as described in Section 5, and entails neither explicit or implicit segmentation into characters.

In this section, we are concerned with HMMs of type 2, which model sequences of feature values obtained from individual letters. For example, Fig. 13 shows a sample feature vector produced from the word "cat." This sequence can be segmented into three letters in many different ways, of which two are shown. The probability that a particular segmentation resulted from the word "cat" is the product of the probabilities of segment 1 resulting from "c," segment 2 from "a," etc. The probability of a different lexicon word can likewise be calculated. To choose the most likely word from a set of alternatives the designer of the system may select either the composite model that gives the segmentation having greatest probability, or else that model which maximizes the a posteriori probability of the observations, i.e., the sum over all segmentations. In either case, the optimization algorithm is organized to avoid redundant calculations, in the former case, by using the well-known Viterbi algorithm.

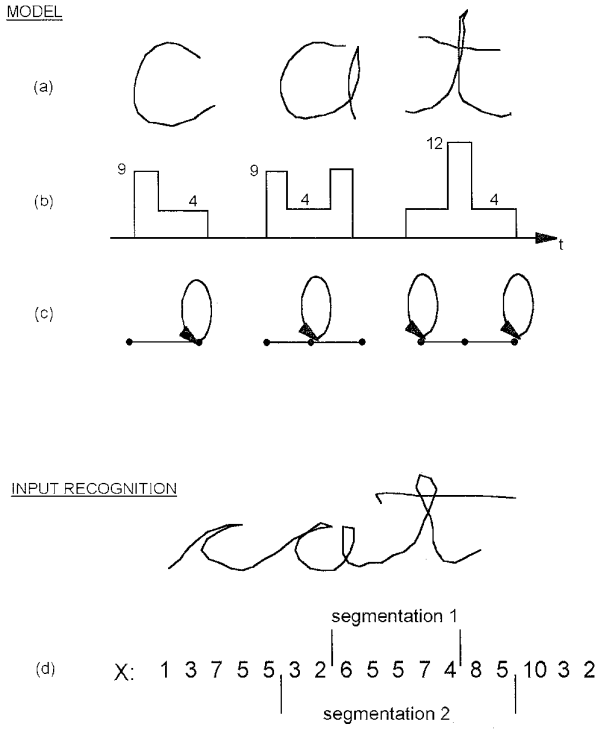


Fig. 13. Hidden Markov Models. This vastly oversimplified example is intended only to illustrate the principal concepts: (a) Training letters and (b) typical sequences of feature values obtained from (a). (c) The Markov models underlying (b), indicating the state sequence, and showing that certain states may be re-entered. Each state outputs a value from a feature distribution whose mean is indicated in the diagram above. The model for a word is the concatenation of such letter models. (d) A sequence of feature values obtained from a word, indicating several different segmentations. The HMM solution is found by evaluating many possible segmentation, of which two are shown.

Such HMMs are a powerful tool to model the fact that letters do not always have distinct segmentation boundaries. It is clear that in the general case perfect letter dissection can not be achieved. This problem can be compensated by the HMMs, as they are able to learn by observing letter segmentation behavior on a training set. Context (word and letter frequencies, syntactic rules) can be also be included, by defining transition probabilities between letter states.

Elementary HMMs describing letters can be combined to form either several model-discriminant word HMMs or else a single path-discriminant model. In model-discriminant HMMs, one model is constructed for each different word

[32], [15], [75], [4] while in the path discriminant HMM only one global model is constructed [50]. In the former case, each word model is assessed to determine which is most likely to have produced a given set of observations. In the latter case, word recognition is performed by finding the most likely paths through the unique model, each path being equivalent to a sequence of letters. Path discriminant HMMs can handle large vocabularies, but are generally less accurate than model-discriminant HMMs. They may incorporate a lexicon comparison module in order to ignore invalid letter sequences obtained by path optimization.

Calculation of the best paths in the HMM model is usually done by means of the Viterbi algorithm. Transition and observed feature probabilities can be learned using the Baum-Welch algorithm. Starting from an initial evaluation, HMM probabilities can be re-estimated using frequencies of observations measured on the training set [33].

First order Markov models are employed in most applications; in [50], an example of a second order HMM is given. Models for cursive script ordinarily assume discrete feature values. However, continuous probability densities may also be used, as in [3].

3.3.2 Non-Markov Approaches

A method stemming from concepts used in machine vision for recognition of occluded objects is reported in [16]. Here, various features and their positions of occurrence are recorded for an image. Each feature contributes an amount of evidence for the existence of one or more characters at the position of occurrence. The positions are quantized into bins such that the evidence for each character indicated in a bin can be summed to give a score for classification. These scores are subjected to contextual processing using a predefined lexicon in order to recognize words. The method is being applied to text printed in a known proportional font.

A method that recognizes word feature graphs is presented in [71]. This system attempts to match subgraphs of features with predefined character prototypes. Different alternatives are represented by a directed network whose nodes correspond to the matched subgraphs. Word recognition is performed by searching for the path that gives the best interpretation of the word features. The characters are detected in the order defined by the matching quality. These can overlap or can be broken or underlined.

This family of recognition-based approaches has more often been aimed at cursive handwriting recognition. Probabilistic relaxation was used in [37] to read off-line handwritten words. The 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 illegible combinations of letters. A major drawback of this technique is that it requires intensive computation.

Tappert employed Elastic Matching to match the drawing of an unknown cursive word with the possible sequences of letter prototypes [80]. 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 that permitted the process to skip unnecessary features. Digram statistics and segmentation constraints were eventually added to improve performance.

Several authors proposed a Hypothesis Testing and Verification scheme to recognize handprinted [44] or on-line cursive words [5], [67]. For example, in the system proposed in [5] a sequence of structural features (like x- and y-extrema, curvature signs, cusps, crossings, penlifts, and closures) was extracted from the word to generate all the legible sequences of letters. Then, the "aspect" of the word (which was deduced from ascender and descender detection) was taken into account to chose the best solution(s) among the list of generated words. In [67], words and letters were represented by means of tree dictionaries: Possible words were described by a letter tree (also called a "trie") and letters were described by a feature tree. The letters were predicted by finding in the letter tree the paths compatible with the extracted features and were verified by checking their compatibility with the word dictionary.

Hierarchical grouping of on-line features was proposed in [39]. The words were described by means of a hierarchical description where primitive features were progressively grouped into more sophisticated representations. The first level corresponded to the "turning points" of the drawing, the second level was dealing with more sophisticated features called "primary shapes," and finally, the third level was a trellis of tentative letters and ligatures. Ambiguities were 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 [77]. In this system, a stroke graph representing the word is obtained after skeletonization. The "singular parts," which are supposed to convey most of the information, were deduced by eliminating "regular part" of the word (the sinusoid-like path joining all cursive ligatures). The most robust features and characters (the "anchors") were then detected from a description chain derived from these singular parts and dynamic matching was used for analyzing the remaining parts.

A top-down directed word verification method called "backward matching" (see Fig. 14) is proposed in [54]. 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, as in human vision [82]. 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 pro-

vide a general framework allowing efficient cooperation between low-level and high-level recognition processes.

4 Mixed Strategies: "Oversegmenting"

Two radically different segmentation strategies have been considered to this point. One (Section 2) attempts to choose the correct segmentation points (at least for generating graphemes) by a general analysis of image features. The other strategy (Section 3) is at the opposite extreme. No dissection is carried out. Classification algorithms simply do a form of model-matching against image contents.

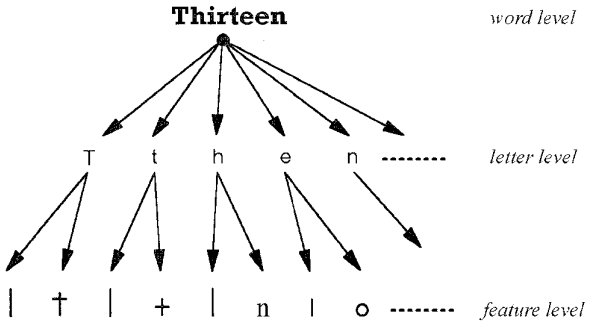


Fig. 14. Backward matching. Recognition is performed by matching an image against various candidate words, the most distinctive letters being matched first. In this example, the algorithm first seeks the letter "T" in the image, then "t," "h," and so on. These letters are more informative visually because they contain ascenders, and lexically because they are consonants.

In this section, intermediate approaches, essentially hybrids of the first two, are discussed. This family of methods also uses presegmenting, with requirements that are not as strong as in the grapheme approach. A dissection algorithm is applied to the image, but the intent is to "oversegment," i.e., to cut the image in sufficiently many places that the correct segmentation boundaries are included among the cuts made, as in Fig. 15. Once this is assured, the optimal segmentation is defined by a subset of the cuts made. Each subset implies a segmentation hypothesis, and classification is brought to bear to evaluate the different hypotheses and choose the most promising segmentation.

Fig. 15. Oversegmenting. Note that letters that contain valleys have been dissected into multiple parts. However, no merged characters remain, so that a correct segmentation can be produced by recombination of some of the segments.

The strategy in a simple form is illustrated in [29]. Here, a great deal of effort was expended in analyzing the shapes of pairs of touching digits in the neighborhood of contact, leading to algorithms for determining likely separation boundaries. However, multiple separating points were tested, i.e., the touching character pair was oversegmented. Each candidate segmentation was tested separately by classification, and the split giving the highest recognition confidence was accepted. This approach reduced segmentation errors 100-fold compared with the previously used segmentation technique that did not employ recognition confidence.

Because touching was assumed limited to pairs, the above method could be implemented by splitting a single image along different cutting paths. Thus, each segmentation hypothesis was generated in a single step. When the number of characters in the image to be dissected is not known a priori, or if there are many touching characters, e.g., cursive writing, then it is usual to generate the various hypotheses in two steps. In the first step, a set of likely cutting paths is determined, and the input image is divided into elementary components by separating along each path. In the second step, segmentation hypotheses are generated by forming combinations of the components. All combinations meeting certain acceptability constraints (such as size, position, etc.) are produced and scored by classification confidence. An optimization algorithm, typically implemented on dynamic programming principles and possibly making use of contextual knowledge, does the actual selection.

A number of researchers began using this basic approach at about the same time, e.g., [46], [9], [13]. Lexical matching is included in the overall process in [26] and [78].

It is also possible to carry out an oversegmenting procedure sequentially by evaluating trial separation boundaries [2]. In this work, a neural net was trained to detect likely cutting columns for machine printed characters using neighborhood characteristics. Using these as a base, the optimization algorithm recursively explored a tree of possible segmentation hypotheses. The left column was fixed at each step, and various right columns were evaluated using recognition confidence. Recursion is used to vary the left column as well, but pruning rules are employed to avoid testing all possible combinations.

5 Holistic Strategies

A holistic process recognizes an entire word as a unit. A major drawback of this class of methods is that their use is usually restricted to a predefined lexicon: Since they do not deal directly with letters but only with words, recognition is necessarily constrained to a specific lexicon of words. This point is especially critical when training on word samples is required: 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 as well as to the particular vocabulary concerned.

Whole word recognition was introduced by Earnest at the beginning of the 1960s [21]. 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 and no temporal information was

used. Recognition was based on the comparison of a collection of simple features extracted from the whole word against 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 and ascenders and descenders were found by considering the part of the writing exceeding this zone. The lexicon of possible word codes was obtained by means of a transcoding table describing all the usual ways of writing letters.

This strategy still typifies recent holistic methods. Systems still use middle zone determination to detect the ascenders and descenders. The type of extracted features also remains globally the same (ascenders, descenders, directional strokes, cusps, diacritical marks, etc.). Finally, holistic methods, as illustrated in Fig. 16, usually 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.

Thus, conceptually, holistic methods use the "classical approach" defined in Section 1, with complete words as the symbols to be recognized. The main advances in recent techniques reside in the way comparison between hypotheses and references is performed. Recent comparison techniques are more flexible and better take into account the dramatic variability of handwriting. These techniques (which were originally introduced for speech recognition) are generally based on Dynamic Programming with optimization criteria based either on distance measurements or 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.

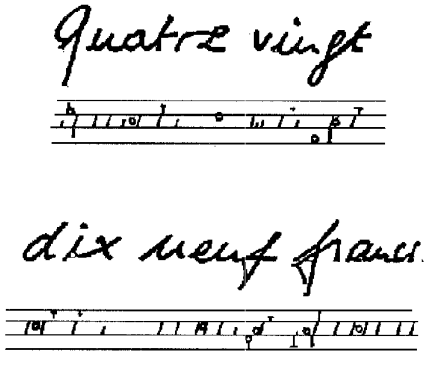


Fig. 16. Holistic recognition. Recognition consists of comparing a lexicon of word descriptions against a sequence of features obtained from an unsegmented word image. The detected features, shown below the images, include loops, oriented strokes, ascenders and descenders.

Dynamic Programming was employed in [62] and [70] for check and city name recognition. Words were represented by a list of features indicating the presence of ascenders, descenders, directional strokes and closed loops. The "middle zone" was not delimited by straight lines, but by means of smooth curves following the central part of the word, even if slanted or irregular in size. Relative y-location was associated to every feature and uncertainty coefficients were introduced to make this representation more tolerant to distortion by avoiding binary decisions. A similar scheme was used in [68] and [56], but with different features. In the first case, features were based on the notion of "guiding points," (which are the intersection of the letters and the median line of the word), whereas in the latter case they are derived from the contours of words.

One of the first systems using Markov Models was developed by Farag in 1979 [25]. In this method, each word is seen as a sequence of oriented strokes which are coded 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 and each matrix contains the transition probabilities from the jth stroke to the following one. The recognized word is the reference Wi of the lexicon which maximizes the joint probability P(Z, Wi) where Z is the unknown word.

Hidden Markov Models are used in [63] for the recognition of literal digits and in [33] for off-line cheque recognition. Angular representation is used in the first system to represent the feature, while structural off-line primitives are used in the second case. Moreover, this second system also implement several Markov models 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 checks was recently proposed in [35]. In the learning stage, lines are extracted from binary images of words and accumulated in prototypes called "holographs." 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 is represented in the prototype space. Each class is approximated with a Gaussian density inside this space and these densities are used to calculate the probability that the word belongs to each class. Other simple holistic features (ascenders and descenders, loops, length of the word) are also used in combination with this main method.

In the machine-printed text area characters are regular so that feature representations are stable, and in a long document repetitions of the most common words occur with predictable frequency. In [45], these characteristics were combined to cluster the ten most common short words with good accuracy, as a precursor to word recognition. It was suggested that identification of the clusters could be done on the basis of unigram and bigram frequencies.

More general applications require a dynamic generation stage of holistic descriptions. This stage converts words from ASCII form to the holistic representation required by the recognition algorithm. Word representation is generated from generic information about letter and ligature representations using a reconstruction model. Word reconstruction is required by applications dealing with a dynamically defined lexicon, for instance the postal application [60] where the list of possible city names is derived

from zip code recognition. Another interesting characteristic of this last technique is that it is not used to find "the best solution" but to filter the lexicon by reducing its size (a different technique being then used to complete recognition). The system was able to achieve 50% size reduction with under 2% error.

6 Concluding Remarks

Methods for treating the problem of segmentation in character recognition have developed remarkably in the last decade. A variety of techniques has emerged, influenced by developments in related fields such as speech and online recognition. In this paper, we have proposed an organization of these methods under three basic strategies, with hybrid approaches also identified. It is hoped that this comprehensive discussion will provide insight into the concepts involved, and perhaps provoke further advances in the area.

The difficulty of performing accurate segmentation is determined by the nature of the material to be read and by its quality. Generally, missegmentation rates for unconstrained material increase progressively from machine print to handprint to cursive writing. Thus, simple techniques based on white separations between characters suffice for clean fixed-pitch typewriting. For cursive script from many writers and a large vocabulary, at the other extreme, methods of ever increasing sophistication are being pursued. Current research employs models not only of characters, but also words and phrases, and even entire documents, and powerful tools such as HMM, neural nets, contextual methods are being brought to bear. While we have focused on the segmentation problem it is clear that segmentation and classification have to be treated in an integrated manner to obtain high reliability in complex cases.

The paper has concentrated on an appreciation of principles and methods. We have not attempted to compare the effectiveness of algorithms, or to discuss the crucial topic of evaluation. In truth, it would be very difficult to assess techniques separate from the systems for which they were developed. We believe that wise use of context and classifier confidence has led to improved accuracies, but there is little experimental data to permit an estimation of the amount of improvement to be ascribed to advanced techniques. Perhaps with the wider availability of standard databases, experimentation will be carried out to shed light on this issue.

We have included a list of references sufficient to provide a more-detailed understanding of the approaches described. We apologize to researchers whose important contributions may have been overlooked.

Acknowledgments

An earlier, abbreviated version of this survey was presented at ICDAR 95 in Montreal, Canada. Prof. George Nagy and Dr. Jianchang Mao read early drafts of the paper and offered critical commentaries that have been of great use in the revision process. However, to the authors falls full responsibility for faults of omission or commission that remain.

Richard G. Casey's research for this paper was performed during his sabbatical at École Nationale Supérieure des Télécommunications in Paris.

References

[1] H.S. Baird, S. Kahan, and T. Pavlidis, "Components of an Omnifont Page Reader," *Proc. Eighth Int'l Conf. Pattern Recognition*, Paris, pp. 344-348, 1986.

[2] T. Bayer, U. Kressel, and M. Hammelsbeck, "Segmenting Merged Characters," *Proc. 11th Int'l Conf. Pattern Recognition*, vol. 2. conf. B: Pattern Recognition, Methodology, and Systems, pp. 346-349, 1992.

[3] E.J. Bellegarda, J.R. Bellegarda, D. Nahamoo, and K.S. Nathan, "A Probabilistic Framework for On-line Handwriting Recognition," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 225, May 1993.

[4] S. Bercu and G. Lorette, "On-line Handwritten Word Recognition: An Approach Based on Hidden Markov Models," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 385, May 1993.

[5] M. Berthod and S. Ahyan, "On Line Cursive Script Recognition: A Structural Approach with Learning," *Proc. Fifth Int'l Conf. Pattern Recognition*, p. 723, 1980.

[6] M. Bokser, "Omnidocument Technologies," *Proc. IEEE*, vol. 80, no. 7, pp. 1,066-1,078, July 1992.

[7] R. Bozinovic and S.N. Srihari, "String Correction Algorithm for Cursive Script Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 4, no. 6, pp. 655-663, June 1982.

[8] R.M. Bozinovic and S.N. Srihari, "Off-Line Cursive Script Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, no. 1, p. 68, Jan. 1989.

[9] T. Breuel, "Design and Implementation of a System for Recognition of Handwritten Responses on US Census Forms," *Proc. IAPR Workshop Document Analysis Systems*, Kaiserlautern, Germany, Oct. 1994.

[10] C.J.C. Burges, J.I. Be, and C.R. Nohl, "Recognition of Handwritten Cursive Postal Words using Neural Networks," *Proc. USPS Fifth Advanced Technology Conf.*, p. A-117, Nov./Dec. 1992.

[11] R.G. Casey and G. Nagy, "Recursive Segmentation and Classification of Composite Patterns," *Proc. Sixth Int'l Conf. Pattern Recognition*, p. 1,023, 1982.

[12] R.G. Casey, "Text OCR by Solving a Cryptogram," *Proc. Eighth Int'l Conf. Pattern Recognition*, Paris, pp. 349-351, Oct. 1986.

[13] R.G. Casey, "Segmentation of Touching Characters in Postal Addresses," *Proc. Fifth US Postal Service Technology Conf.*, Washington D.C., 1992.

[14] M. Cesar and R. Shinghal, "Algorithm for Segmenting Handwritten Postal Codes," *Int'l J. Man Machine Studies*, vol. 33, no. 1, pp. 63-80, July 1990.

[15] M.Y. Chen and A. Kundu, "An Alternative to Variable Duration HMM in Handwritten Word Recognition," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 82, May 1993.

[16] C. Chen and J. DeCurtins, "Word Recognition in a Segmentation-Free Approach to OCR," *Proc. Int'l Conf. Document Analysis and Recognition*, Tsukuba City, Japan, pp. 573-576, Oct. 1993.

[17] M. Cheriet, Y.S. Huang, and C.Y. Suen, "Background Region-Based Algorithm for the Segmentation of Connected Digits," *Proc. 11th Int'l Conf. Pattern Recognition*, vol. 2, p. 619, Sept. 1992.

[18] M. Cheriet, "Reading Cursive Script by Parts," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 403, May 1993.

[19] G. Dimauro, S. Impedovo, and G. Pirlo, "From Character to Cursive Script Recognition: Future Trends in Scientific Research," *Proc. 11th Int'l Conf. Pattern Recognition*, vol. 2, p. 516, Aug. 1992.

[20] C.E. Dunn and P.S.P. Wang, "Character Segmenting Techniques for Handwritten Text—A Survey," *Proc. 11th Int'l Conf. Pattern Recognition*, vol. 2, p. 577, Aug. 1992.

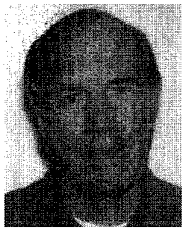
[21] L.D. Earnest, "Machine Recognition of Cursive Writing," C. Cherry, ed., *Information Processing*, pp. 462-466. London: Butterworth, 1962.

[22] R.W. Ehrich and K.J. Koehler, "Experiments in the Contextual Recognition of Cursive Script," *IEEE Trans. Computers*, vol. 24, no. 2, p. 182, Feb. 1975.

[23] D.G. Elliman and I.T. Lancaster, "A Review of Segmentation and Contextual Analysis Techniques for Text Recognition," *Pattern Recognition*, vol. 23, no. 3/4, pp. 337-346, 1990.

[24] R.J. Evey, "Use of a Computer to Design Character Recognition Logic," *Proc. Eastern Joint Computer Conf.*, pp. 205-211, 1959.
[25] R.F.H. Farag, "Word-Level Recognition Recognition of Cursive Script," *IEEE Trans. Computers*, vol. 28, no. 2, pp. 172-175, Feb. 1979.
[26] J.T. Favata and S.N. Srihari, "Recognition of General Handwritten Words Using a Hypothesis Generation and Reduction Methodology," *Proc. Fifth USPS Advanced Technology Conf.*, p. 237, Nov./Dec. 1992.
[27] R. Fenrich, "Segmenting of Automatically Located Handwritten Numeric Strings," *From Pixels to Features III*, S. Impedovo and J.C. Simon, eds., Chapter 1, p. 47, Elsevier, 1992.
[28] P.D. Friday and C.G. Leedham, "A Pre-Segmenter for Separating Characters in Unconstrained Hand-Printed Text," *Proc. Int'l Conf. Image Proc.*, Singapore, Sept. 1989.
[29] H. Fujisawa, Y. Nakano, and K. Kurino, "Segmentation Methods for Character Recognition: From Segmentation to Document Structure Analysis," *Proc. IEEE*, vol. 80, no. 7, pp. 1,079-1,092, July 1992.
[30] K. Fukushima and T. Imagawa, "Recognition and Segmentation of Connected Characters with Selective Attention," *Neural Networks*, vol. 6, no. 1, pp. 33-41, 1993.
[31] P. Gader, M. Magdi, and J-H. Chiang, "Segmentation-Based Handwritten Word Recognition," *Proc. USPS Fifth Advanced Technology Conf.*, Nov./Dec. 1992.
[32] A.M. Gillies, "Cursive Word Recognition Using Hidden Markov Models," *Proc. USPS Fifth Advanced Technology Conf.*, Nov./Dec. 1992.
[33] M. Gilloux, J.M. Bertille, and M. Leroux, "Recognition of Handwritten Words in a Limited Dynamic Vocabulary," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 417, 1993.
[34] M. Gilloux, "Hidden Markov Models in Handwriting Recognition," *Fundamentals in Handwriting Recognition*, S. Impedovo, ed., NATO ASI Series F: Computer and Systems Sciences, vol. 124, Springer Verlag, 1994.
[35] N. Gorsky, "Off-line Recognition of Bad Quality Handwritten Words Using Prototypes," *Fundamentals in Handwriting Recognition*, S. Impedovo, ed., NATO ASI Series F: Computer and Systems Sciences, vol. 124, Springer Verlag, 1994.
[36] L.D. Harmon, "Automatic Recognition of Print and Script," *Proc. IEEE*, vol. 60, no. 10, pp. 1,165-1,177, Oct. 72.
[37] K.C. Hayes, "Reading Handwritten Words Using Hierarchical Relaxation," *Computer Graphics and Image Processing*, vol. 14, pp. 344-364, 1980.
[38] R.B. Hennis, "The IBM 1975 Optical Page Reader: System Design," *IBM J. Research and Development*, pp. 346-353, Sept. 1968.
[39] C.A. Higgins and R. Whitrow, "On-Line Cursive Script Recognition," *Proc. Int'l Conf. Human-Computer Interaction—INTERACT '84*, Elsevier, 1985.
[40] W.H. Highleyman, "Data for Character Recognition Studies," *IEEE Trans. Electrical Computation*, pp. 135-136, Mar. 1963.
[41] 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, p. 821, 1992.
[42] M. Holt, M. Beglou, and S. Datta, "Slant-Independent Letter Segmentation for Off-line Cursive Script Recognition," *From Pixels to Features III*, S. Impedovo and J.C. Simon, eds., p. 41, Elsevier, 1992.
[43] R.L. Hoffman and J.W. McCullough, "Segmentation Methods for Recognition of Machine-Printed Characters," *IBM J. Research and Development*, pp. 153-65, Mar. 1971.
[44] J.J. Hull and S.N. Srihari, "A Computational Approach to Visual Word Recognition: Hypothesis Generation and Testing," *Proc. Computer Vision and Pattern Recognition*, pp. 156-161, June 1986.
[45] J. Hull, S. Khoubyari, and T.K. Ho, "Word Image Matching as a Technique for Degraded Text Recognition," *Proc. 11th Int'l Conf. Pattern Recognition*, vol. 2, conf. B, pp. 665-668, Sept. 1992.
[46] F. Kimura, S. Tsuruoka, M. Shridhar, and Z. Chen, "Context-Directed Handwritten Word Recognition for Postal Service Applications," *Proc. Fifth US Postal Service Technology Conf.*, Washington, D.C., 1992.
[47] F. Kimura, M. Shridhar, and N. Narasimhamurthi, "Lexicon Directed Segmentation-Recognition Procedure for Unconstrained Handwritten Words," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 122, May 1993.
[48] V.A. Kovalevsky, *Character Readers and Pattern Recognition*. Washington, D.C.: Spartan Books, 1968.
[49] F. Kuhl, "Classification and Recognition of Hand-Printed Characters," *IEE Nat'l Convention Record*, pp. 75-93, Mar. 1963.
[50] A. Kundu, Y. He, and P. Bahl, "Recognition of Handwritten Words: First and Second Order Hidden Markov Model Based Approach," *Pattern Recognition*, vol. 22, no. 3, p. 283, 1989.
[51] E. Lecolinet and J-V. Moreau, "A New System for Automatic Segmentation and Recognition of Unconstrained Zip Codes," *Proc. Sixth Scandinavian Conf. Image Analysis*, Oulu, Finland, p. 585, June 1989.
[52] E. Lecolinet, "Segmentation d'images de mots manuscrits," PhD thesis, Université Pierre et Marie Curie, Paris, Mar. 1990.
[53] E. Lecolinet and J-P. Crettez, "A Grapheme-Based Segmentation Technique for Cursive Script Recognition," *Proc. Int'l Conf. Document Analysis and Recognition*, Saint Malo, France, p. 740, Sept. 1991.
[54] E. Lecolinet, "A New Model for Context-Driven Word Recognition," *Proc. Symp. Document Analysis and Information Retrieval*, Las Vegas, p. 135, Apr. 1993.
[55] E. Lecolinet and O. Baret, "Cursive Word Recognition: Methods and Strategies," *Fundamentals in Handwriting Recognition*, S. Impedovo, ed., NATO ASI Series F: Computer and Systems Sciences, vol. 124, pp. 235-263, Springer Verlag, 1994.
[56] M. Leroux, J-C. Salome, and J. Badard, "Recognition of Cursive Script Words in a Small Lexicon," *Int'l Conf. Document Analysis and Recognition*, Saint Malo, France, p. 774, Sept. 1991.
[57] S. Liang, M. Ahmadi, and M. Shridhar, "Segmentation of Touching Characters in Printed Document Recognition," *Proc. Int'l Conf. Document Analysis and Recognition*, Tsukuba City, Japan, pp. 569-572, Oct. 1993.
[58] G. Lorette and Y. Lecourtier, "Is Recognition and Interpretation of Handwritten Text: A Scene Analysis Problem?" *Pre-Proc. IWFHR III*, p. 184, Buffalo, N.Y., May 1993.
[59] Y. Lu, "On the Segmentation of Touching Characters," *Int'l Conf. Document Analysis and Recognition*, Tsukuba, Japan, pp. 440-443, Oct. 1993.
[60] S. Madhvanath and V. Govindaraju, "Holisitic Lexicon Reduction," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 71, May 1993.
[61] M. Maier, "Separating Characters in Scripted Documents," *Proc. Eighth Int'l Conf. Pattern Recognition*, Paris, p. 1,056, 1986.
[62] J-V. Moreau, B. Plessis, O. Bourgeois, and J-L. Plagnaud, "A Postal Check Reading System," *Int'l Conf. Document Analysis and Recognition*, Saint Malo, France, p. 758, Sept. 1991.
[63] R. Nag, K.H. Wong, and F. Fallside, "Script Recognition Using Hidden Markov Models," *IEEE ICASSP*, Tokyo, pp. 2,071-2,074, 1986.
[64] T. Nartker, *ISRI 1992 Annual Report*, Univ. of Nevada, Las Vegas, 1992.
[65] T. Nartker, *ISRI 1993 Annual Report*, Univ. of Nevada, Las Vegas, 1993.
[66] K. Ohta, I. Kaneko, Y. Itamoto, and Y. Nishijima, *Character Segmentation of Address Reading/Letter Sorting Machine for the Ministry of Posts and Telecommunications of Japan*, NEC Research and Development, vol. 34, no. 2, pp. 248-256, Apr. 1993.
[67] H. Ouladj, G. Lorette, E. Petit, J. Lemoine, and M. Gaudaire, "From Primitives to Letters: A Structural Method to Automatic Cursive Handwriting Recognition," *Proc. Sixth Scandinavian Conf. Image Analysis*, Finland, p. 593, June 1989.
[68] T. Paquet and Y. Lecourtier, "Handwriting Recognition: Application on Bank Cheques," *Proc. Int'l Conf. Document Analysis and Recognition*, Saint Malo, France, p. 749, Sept. 1991.
[69] S.K. Parui, B.B. Chaudhuri, and D.D. Majumder, "A Procedure for Recognition of Connected Handwritten Numerals," *Int'l J. Systems Science*, vol, 13, no. 9, pp. 1,019-1,029, 1982.
[70] B. Plessis, A. Sicsu, L. Heute, E. Lecolinet, O. Debon, and J-V. Moreau, "A Multi-Classifier Strategy for the Recognition of Handwritten Cursive Words," *Proc. Int'l Conf. Document Analysis and Recognition*, Tsukuba City, Japan, pp. 642-645, Oct. 1993.
[71] J. Rocha and T. Pavlidis, "New Method for Word Recognition Without Segmentation," *Proc. SPIE Character Recognition Technologies*, vol. 1,906, pp. 74-80, 1993.
[72] K.M. Sayre, "Machine Recognition of Handwrittten Words: A Project Report," *Pattern Recognition*, vol. 5, pp. 213-228, 1973.
[73] J. Schuermann, "Reading Machines," *Proc. Sixth Int'l Conf. Pattern Recognition*, Munich, Germany, 1982.
[74] G. Seni and E. Cohen, "External Word Segmentation of Off-Line Handwritten Text Lines," *Pattern Recognition*, vol. 27, no. 1, pp. 41-52, Jan. 1994.

[75] A.W. Senior and F. Fallside, "An Off-line Cursive Script Recognition System Using Recurrent Error Propagation Networks," *Pre-Proc. IWFHR III*, Buffalo, N.Y., p. 132, May 1993.
[76] M. Shridhar and A. Badreldin, "Recognition of Isolated and Simply Connected Handwritten Numerals," *Pattern Recognition*, vol. 19, no. 1, p. 1, 1986.
[77] J.C. Simon, "Off-line Cursive Word Recognition," *Proc. IEEE*, p. 1,150, July 1992.
[78] R.M.K Sinha, B. Prasada, G. Houle, and M. Sabourin, "Hybrid Recognition with String Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 915-925, Sept. 1993.
[79] M.E. Stevens, "Automatic Character Recognition—A State of the Art Report," MES National Bureau of Standards Technical Note no. 112, 1961.
[80] C.C. Tappert, "Cursive Script Recognition by Elastic Matching," *IBM J. Research Development*, vol. 26, pp. 765-771, Nov. 1982.
[81] C.C. Tappert, C.Y. Suen, and T. Wakahara, "The State of the Art in On-line Handwriting Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, p. 787, Aug. 1990.
[82] I. Taylor and M. Taylor, *The Psychology of Reading*. Academic Press, 1983.
[83] S. Tsujimoto and H. Asada, "Major Components of a Complete Text Reading System," *Proc. IEEE*, vol. 80, no. 7, pp. 1,133-1,149, July 1992.
[84] J. Wang and J. Jean, "Segmentation of Merged Characters by Neural Networks and Shortest Path," *Pattern Recognition*, vol. 27, no. 5, pp. 649-658, May 1994.
[85] J.M. Westall and M.S. Narasimha, "Vertex Directed Segmentation of Handwritten Numerals," *Pattern Recognition*, vol. 26, no. 10, pp. 1,473-1,186, Oct. 1993.
[86] R.A. Wilkinson, *Proc. First Conf. Census Optical Character Recognition System*, Report No. PB92-238542/XAB, National Institute of Standards and Technology, Gaithersburg, Md., May 1992.
[87] R.A. Wilkinson, "Comparison of Massively Parallel Segmenters," National Institute of Standards and Technology technical report, Gaithersburg, Md., Sept. 1992.
[88] R.A. Wilkinson, *Proc. Second Conf. Census Optical Character Recognition Systems*, National Institute of Standards and Technology, Gaithersburg, Md., Feb. 1993.
[89] B.A. Yanikoglu and P.A. Sandon, "Recognizing Off-Line Cursive Handwriting," *Proc. Computer Vision and Pattern Recognition*, 1994.



Richard G. Casey received his Doctor of Engineering Science degree from Columbia University in 1965. He had been employed since 1963 at the IBM Research Division, Yorktown Heights, New York, while preparing his thesis on character recognition. While with IBM, he had the good fortune to work with many of the pioneers in pattern recognition. In 1968, he took a leave of absence from IBM to teach pattern recognition and other subjects at the University of Florida. When he returned to IBM in 1970, he transferred to the company's research facility (now the Almaden Research Center) in San Jose, California. In 1975, he helped organize Almaden's document recognition project; it has been continuously active since then.

Dr. Casey's research into OCR, forms data extraction, and related areas has led directly to a number of IBM products. For this research, he has been recognized with IBM awards four times. He was a member of the ISRI Advisory Board for the University of Nevada at Las Vegas in 1992-1995 and program chairman for document analysis at the 1993 symposium. He was an invited speaker at the First ICDAR, and has been a program committee member for numerous conferences on document analysis. When he retired from IBM in 1994, he went on a six-month sabbatical at École Nationale Supérieure des Télécommunications in Paris.

Eric Lecolinet received his PhD degree in computer science from Université Pierre et Marie Curie, Paris, in 1990. He worked on optical character recognition and cursive script recognition in Matra, France, from 1987 to 1990, and at the IBM Almaden Research Center, San Jose, California, from 1990 to 1992. He is now an associate professor at École Nationale Supérieure des Télécommunications in Paris. His current research interests include pattern recognition, artificial intelligence, and human-computer interaction. He is a member of the IEEE.