POSTAL SERVICES

> On July 26, 1775, members of the Second Continental Congress, meeting in Philadelphia, agreed . . . that a Postmaster General be appointed for the United States, who shall hold his office at Philadelphia, and shall be allowed a salary of 1,000 dollars per annum . . ..

This meeting signaled the birth of the Post Office Department, which subsequently became the United States Postal Service (USPS) (information provided here comes directly from the Web pages of US Postal Service). It is interesting to note that the USPS is one of the oldest agencies of the United States of America.

Mail delivery has evolved significantly since the days of horseback, stagecoach, steamboat, railroad, automobile, and airplane. Mail contracts generated the income necessary to build the great highways, rail lines, and airways that now span the continent.

ZIP Codes. The tremendous increase in mail volume, the steep rise in labor costs, and the revolution in transportation led to rapid adoption of modern technology resulting in the ZIP (Zoning Improvement Plan) code. By July 1963, a five-digit code had been assigned to every address throughout the United States. In a five-digit ZIP code, the first digit designates a broad geographical area of the United States, ranging from zero for the Northeast to nine for the far West. This was followed by two digits that pinpointed population concentrations and those centers accessible to transportation networks.

The final two digits designated small post offices or postal zones in larger zoned cities. Thus a ZIP code of 48323 would identify the West Bloomfield Township in the state of Michigan. In fact the first two digits (48) would identify Michigan.

An important milestone occurred in November 1965 when the Postal Service installed a high-speed optical character reader (OCR) in the Detroit Post Office. This first-generation machine read the city/state/ZIP code line of typed addresses to sort letters to one of the 277 pockets. This automation led to increased productivity. In order to offset rising costs associated with growing mail volume and to reduce the number of mail piece processing, the Postal Service developed an expanded ZIP code in 1978.

ZIP+4. The ZIP+4 code, which was introduced in 1983, added a hyphen and four additional digits to the existing five-digit ZIP code. The first five numbers continued to identify an area of the country and delivery office to which mail is directed. The sixth and seventh numbers denote a delivery sector, which may be several blocks, a group of streets, a group of post office boxes, several office buildings, a single high-rise office building, a large apartment building, or a small geographic area. The last two numbers denote a delivery segment, which might be one floor of an office building, one side of a street between intersecting streets, specific departments in a firm, or a group of post office boxes.

The age of automation was ushered in in September 1982 when the first computer-driven single-line OCR was intalled in Los Angeles. The equipment utilized OCR to read the letter and print a barcode on the envelope. At the destination post office, a less expensive barcode sorter (BCS) sorted the mail by reading its barcode. By the end of 1984, 252 OCRs capable of processing 24,000 pieces of mail per hour were installed in 118 major processing centers across the country with an average productivity rate of 6200 pieces per work hour. This was a substantial increase when compared to the 1750 pieces per work hour processed previously. Currently, USPS has deployed more than 800 multiline optical character readers (MLOCRs) that can read addresses at 40,000 letter pieces per hour and assign corresponding ZIP+4 codes. Of the letters currently fed to the MLOCRs, 15% have handwritten addresses and these are mostly rejected by the reader. Today, a new generation of equipment is changing the way mail flows and improving productivity. MLOCRs read the entire address on an envelope, print a barcode on the envelope, and then sort it at the rate of more than nine per second. Wide area barcode readers can read a barcode located anywhere on a letter. Advanced facer-canceler systems face, cancel, and sort mail. The remote barcoding system (RBCS) provides barcodes for handwritten script mail or mail that cannot be read by OCRS.

The ZIP+4 code has reduced the number of times that a piece of mail needs to be handled and has shortened the time carriers spend casing their mail (placing it in order of delivery). The delivery point barcode, which represents an 11-digit ZIP code, will eliminate the need for carriers to sort mail because mail will arrive in trays at the delivery post office sorted in "walk sequence." The MLOCR reads the barcode and address, then imprints a unique 11-digit delivery point barcode using the Postal Service's National Directory and the last two digits of the street address. The 11-digit code consists of ZIP+4 code with two additional digits that uniquely identifies the addressee. The barcode sorters essentially sort the mail in walk sequence for the mail person to effect efficient delivery.

Although MLOCRs have been deployed throughout the United States, some formidable challenges remain. For example, the same MLOCR machine that is deployed across the nation has a performance that ranges from a low 35% ZIP+4 encoding for places like Queens, NY (which has addresses with hyphens and numeral street names), to a high 65% in places like San Diego (which mainly has long street names and a limited number of high-rise buildings). Today, the main hurdle in total automation is the inability of MLOCRs to handle the handwritten addresses that constitute nearly 15% of all letter pieces (amounting to several million pieces) handled by USPS.

The USPS address database has evolved over the years and has advanced technologies in areas of data gathering, data storage, data validation, and database maintenance. Driven by the need for automation, the level of details found in the database started with a few delivery points in a city to every single delivery point in the United States. That is quite an achievement.

MACHINE RECOGNITION OF HANDWRITTEN ADDRESSES

The process by which people recognize handwritten characters, words, and documents has been the subject of intense interest and investigation by researchers from very diverse fields. A good understanding of the mechanism of human recognition of handwritten documents will have a significant impact on the development of machines capable of recognition and interpretation of handwritten documents. However, the human recognition process is quite complex, and it incorporates information extracted at different levels: characters, whole words, key words, and contextual processing. The efficiency of human recognition of handwritting can be attributed to the effective integration of multiple cues and exploitation of redundancies contained in most documents. However, if the goal of this study is to develop machines that are capable of automatic transcription of handwritten documents, then one must recognize the immense difficulty of adopting the human recognition process.

In this article the primary focus will be on the development of practical approaches to handwriting recognition. The word "document" is used in a very general sense. Thus, a document will include characters, words, phrases, sentences, and whole paragraphs. There are two main approaches to handwriting recognition: (1) techniques based on holistic approaches whereby an entire word or a character string is recognized as a unit and (2) techniques based on extraction and recognition of characters (also referred to as segmentation-recognition approach) contained in a word or a string. Due to the focus on practical approaches, this article will present an in-depth overview of recognition techniques based on segmentation-recognition.

This article will be organized as follows:

1. Address interpretation system—an overview
2. Taxonomy
3. Image normalization processes
4. Image presegmentation

5. Context-free recognition of primitives and concatenation of primitives
6. Lexicon-driven recognition based on word matching
7. Case studies

ADDRESS INTERPRETATION SYSTEM—AN OVERVIEW

In this section, the authors present an overview of an integrated handwritten address recognition system that requires detection and recognition of ZIP code field, city/state field, street number field, street name/PO box field, and finally the correct nine-digit ZIP code. The interesting feature of this study is the lack of any a priori information about the nature of the address. Addresses may contain a PO box and/or street number/name fields. The integrated system is required to determine the type of field present and determine the nine-digit ZIP code. Figure 1 illustrates images of handwritten addresses that contain street number and name and/or PO box designation.

Three ZIP code directories are used to generate lists of cities, states, and streets. The five-digit ZIP code directory consists of about 100,000 records containing information about the five-digit ZIP codes of all cities in the United States. The ZIP+4 directory consists of about 26 million records containing information about every street, street number range, and PO box numbers in the United States. The third directory, called the delivery point code (DPC) directory, has over 100 million records that virtually locates any valid address in the United States. Efficient use of these directories is very crucial to successful encoding of mail pieces.

One basic problem in address matching remains, that is, the address in the database is a USPS standard address. Patrons often do not actually write addresses on mail pieces using USPS format.

The handwritten address interpretation system consists of subsystems for preprocessing, ZIP code line recognition, street line recognition, PO box line recognition, and delivery point code determination. Figure 2(a) illustrates the procedure for address interpretation, and Fig. 2(b) shows a typical result. Address interpretation requires the determination of the correct ZIP+4 code by analyzing the image of a given address image. Several preprocessing steps are implemented prior to the critical task of address recognition. The first task is the determination of the destination address block (AB). In this process, one utilizes the fact that destination addresses are generally found in the southeast section of a flat mail piece. By analyzing this region of the image, one may reliably extract the destination AB. The second preprocessing subsystem applies tilt correction, line segmentation, slant correction, and word presegmentation.

The first step in address interpretation is the detection and recognition of the ZIP code field. Again, one uses the common practice of writing the city, state, and the ZIP code in the last line of the address. The ZIP code line recognition subsystem generates several ranked ZIP code candidates.

The street line recognition subsystem generates several ranked pairs of street numbers and street names for given five-digit ZIP code. If the top candidate pair is accepted with sufficient confidence, it is sent to the DPC determination subsystem together with the five-digit ZIP code. The PO box line recognition subsystem generates several ranked PO box numbers for given five-digit ZIP code. If the top candidate is accepted with sufficient confidence, it is sent to the DPC determination subsystem with the five-digit ZIP code.

If the top candidate is a five-digit unique ZIP code with sufficient confidence, it is encoded directly in the DPC determination subsystem. If the top candidate is a nine-digit ZIP code on a mail piece with sufficient confidence, it is also directly encoded to DPC in the DPC determination subsystem. The DPC determination subsystem encodes given information from each subsystem to a DPC. If no valid DPC is obtained and the five-digit ZIP code has sufficiently high confidence, it is accepted. Otherwise it is rejected. Each subsystem is described in subsequent sections.

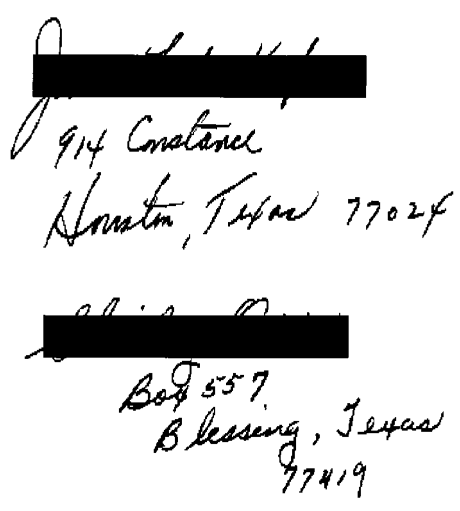

Figure 1. Images of handwritten address blocks (names blocked out for reasons of privacy).

ZIP Code Line Recognition Subsystem

The ZIP code is first assumed to be at the last field of the last line. If the likelihood of the detected ZIP code is less than a threshold, up to two preceding lines are assumed successively to be the ZIP code line until a ZIP code with sufficient likelihood is detected. In actual presegmented images, ZIP code fields are often split and divided into several pieces, which have to be merged again into a field. This problem is resolved through multiple use of the word recognition algorithm to a set of successive presegments. The word recognition algorithm employs a lexicon free word matching described in the section entitled "Word-Matching Algorithm."

Street Line Recognition Subsystem

The street line recognition system consists of three parts:

1. The first part deals with the detection and recognition of the street number field.
2. The second part deals with the generation of a lexicon of street names by accessing the ZIP+4 database with the ZIP code and street and street number as indices.

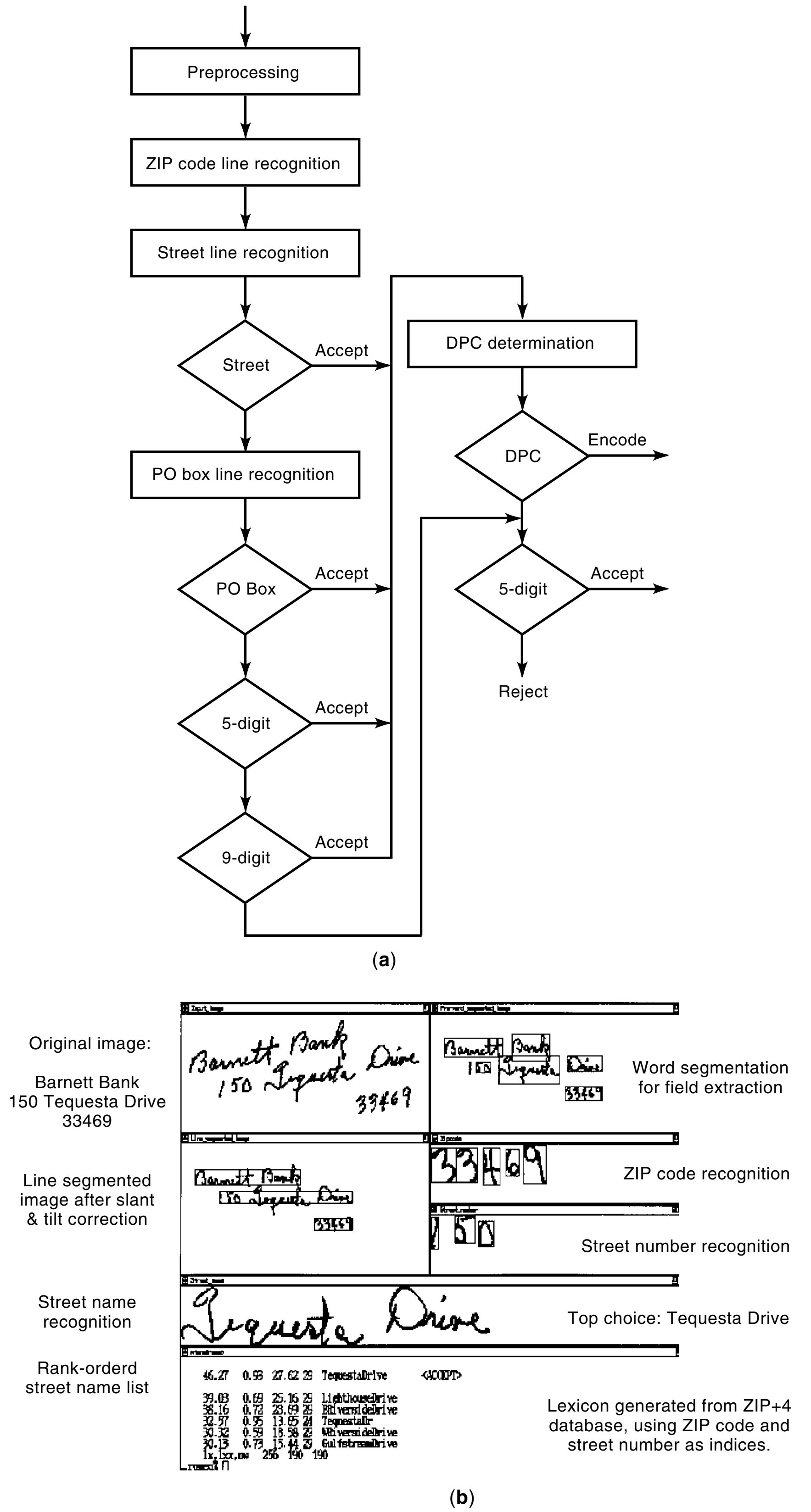

Figure 2. (a) Block diagram for handwritten address interpretation. (b) Result of processing an address image.

3. The third part deals with the recognition of street name through the use of the word-matching algorithm described earlier.

Street Number Location and Recognition. The street number is assumed to be the first field of the street line. If ZIP code line includes only the ZIP code, the second preceding line is first assumed to be the street line; otherwise the immediate preceding line is assumed to be the street line. The rest of the street number location works in the same way as in the ZIP code location. The word recognition algorithm employs a lexicon free word matching.

Street Name Recognition

A street line recognition system is composed ot the ZIP/street number recognition system and the lexicon directed word recognition algorithm. The lexicon is generated through the ZIP+4 directory search for a given pair of ZIP code and street number. The street name recognition is performed in the long word lexicon scheme (i.e., the predirectional, the street name, and the suffix are concatenated in a word) and is dealt as a single word. The word images in a street line except the street number image are supplied as a single word image to the word recognition algorithm. The word recognition algorithm employs a lexicon-directed word matching described in the section entitled "Word-Matching Algorithm."

TAXONOMY

An important goal of this article is to provide a taxonomy of handwriting recognition. In its most general form, handwriting recognition is the transcription of a handwritten document into machine written text. The process starts at the document level and goes through the following steps: (1) extraction of lines from the document, (2) extraction of words from the line, (3) holistic or character-based recognition of words, including punctuation marks such as periods, commas, colons, semicolons, apostrophes, and so on, and (4) a postprocessing step for integration of contextual and a priori knowledge to improve and enhance the recognition process. Figure 1 displays samples of handwritten documents with the characteristics described above.

Techniques for the extraction of lines and extraction of words would be properly classified as preprocessing steps. In general, a handwritten document can have one or more of the following characteristics:

1. *Slant of the Writing.* It is very common to find a distinct slant in the writing habits of most humans. Slant is measured as an angle with respect to a vertical frame of reference.
2. *Skew or Tilt.* A skew occurs when the lines of words are at an angle to the horizontal frame. This often occurs when the writer introduces a skew owing to his/her inability to write on a reference line that is often not physically present.
3. *Underlines.* It is very common to encounter underlines in many handwritten documents. The underlines which are often undulating and not straight are intended to emphasize some key features of the document.
4. *Overlapped Lines.* This is a serious problem in handwriting recognition. Due to limited spacing and the generally poor writing habits of humans, words from one line intrude into adjacent lines, often intersecting with words in the lines above and below.
5. *Discrete Handwriting.* Most people write in a hybrid (neither purely cursive nor discrete) format. A typical handwritten word may have discrete components (single isolated characters) as well as cursive components consisting of several characters. Often these components of a single word are spatially separated. The main problem with discrete writing is the confusion with regard to the location of precise word boundaries in a line of handwritten text.
6. *Imprecise Punctuation.* In handwritten documents, it is often not easy to recognize punctuation marks because they are not precisely rendered during writing. Thus a comma may be mistaken for a character owing to both its size and location.
7. *Broken Characters.* This is a very common occurrence in handwritten documents. Thus the horizontal bar of T may be physically disconnected from the vertical limb. This also occurs for characters such as A, B, D, H, R, and so on.
8. *Similarly Shaped Words.* In handwritten words, it would often be very difficult to distinguish the word "clean" from "dean," when written in a cursive mode. Contextual interpretation would be needed to resolve this type of confusion.
9. *Ligatures.* Again due to the diverse writing styles of people, long and sometimes unusual ligatures connecting adjacent characters often add confusion to the identity of the word. This is especially problematic with words in which one encounters w, u, v, and so on. Other examples include unwanted connections such as in tt and ff.
10. *Overlapping Characters.* Another common occurrence is the overlapping of character fields within a word. Thus a t or a T with a long horizontal limb often overlaps with characters in adjacent positions, making it quite difficult to extract characters. Such overlaps also occur with such combinations as gh, gy, and so on, especially when the bottom loop of g or y is quite wide.

It is in the context of the above observations that one must approach the goal of developing machines capable of reading handwritten documents. It is important to note that even without most of the problems cited above, handwriting recognition would still be a formidable challenge.

Recognition Strategies

Word recognition algorithms may be classified into the following categories: (1) holistic approach or (2) character extraction approach.

The holistic approach generally utilizes shape features extracted from the word image and attempts to recognize the entire word from these features. The character extraction approach segments the word image into primitive components (typically characters). Character segmentation prior to recognition is called external character segmentation, while concurrent segmentation and recognition is called internal character segmentation.

Holistic Approach

In the holistic approach a word is recognized as a unit, and techniques are derived from recognition of the entire word without attempting to analyze the letter content of the word. A set of features (strokes, holes, arcs, ascenders, descenders, etc.) characterizing the entire word is derived and used in recognition. It is generally accepted that holistic methods are feasible only when the number of words to be recognized is small (typically less than 100). One of the earliest papers in holistic recognition is the classic work of Frishkopf and Harmon of Bell Laboratories (1). In this approach a word is represented in terms of its horizontal and vertical extremes. In this context, an extreme is defined as a point at which one finds a horizontal (vertical) maximum or minimum. Thus a word is represented as an ordered list of extremes. Recognition is based on the best match between test features and features derived from dictionary words. Although the test inputs were obtained in an on-line mode, recognition was essentially off-line. Another early paper in holistic recognition was from Earnest (2), who used off-line recognition strategies to on-line test data. Earnest used features extracted from the middle zone of the words, ascenders, and descenders.

As noted earlier, holistic approaches can be used in two principal environments: (1) when the words to be recognized are prespecified and when the number of words is small and (2) when the main objective is the reduction of lexicon size by eliminating obvious mismatches, thereby facilitating a more accurate, but computationally more intensive, technique to be used for final word recognition.

Character Extraction Approach

In this approach, which is also described as a character-based approach, algorithms are derived to extract and recognize the individual characters of the word. There are three principal issues that need to be considered:

1. In cursive writing, it is very difficult, if not impossible, to extract the characters of the word. If one considers words containing the characters w, m, and d, and letter pairs rn, nr, un, iv, and so on, it is evident that many segmentations leading to identifiable characters are possible. Also in cursive writing, it is often difficult to distinguish the letter o from a especially when ligatures are involved.
2. Erroneous recognition of characters extracted from the word image can lead to incorrect word recognition. It is more typical to encounter letter strings that do not constitute a legitimate word. In such cases, it would be necessary to incorporate a postprocessing stage to select the closest words from a lexicon, using expression-matching techniques.
3. The availability of a lexicon of words that also contains the true word is crucial to developing efficient techniques for word recognition. Fortunately in certain applications such as check processing and address recognition, a suitable lexicon can be generated. However, in the case of numeral strings, a lexicon is not usually available.

It is also necessary to recognize that segmentation is not a local process; rather it is dependent on both the previous extracted character (and its identity) and the likely character that follows the current character. Contextually, it is clear that if the current character is q (this character could easily have been recognized as g), then the previous character is most likely a vowel and the following character is, with a probability of 1, the character u. However, if the previous character were g, then the next character would be any legitimate letter. Di-gram and/or tri-gram analysis would be needed to eliminate ambiguous letter strings. This article will describe in detail the word recognition strategies based on segmentation-recognition.

CONCURRENT SEGMENTATION-RECOGNITION OF WORD IMAGES

In this section, word recognition algorithms based on segmentation of a word image into primitive components (characters and subcharacters) and concurrent concatenation and recognition of these primitive components will be described. There are two approaches to word recognition:

1. Context-free recognition where the recognition system yields an optimum letter string and a lexicon is used as a postprocessor to select the best match.
2. Lexicon-directed recognition, where a presegmented word image is matched against all the words of the lexicon to obtain the best match.

While word recognition may be based on context-free or lexicon-directed techniques, numeral string recognition such as ZIP code recognition or street number recognition in an address is predominantly based on context-free techniques. The recognition of words in a document follows a hierarchical scheme as described below:

1. Remove tilt (skew) of the document.
2. Extract lines of words from document.
3. Remove slant from each line.
4. Extract words from each line.
5. Presegmentation of each word into primitive components (characters and subcharacters).
6. Concatenation of components followed by character recognition to recognize each word.

In the section entitled “Recognition Algorith,” lexicon-directed word recognition will be discussed in detail. The extension to context-free recognition will be illustrated.

Preprocessing (Line/Word Segmentation)

In this section we discuss segmentation of lines in a document, segmentation of words in a line, and related techniques such as tilt and slant correction. *Line segmentation* is defined as the process of extracting the individual lines of words from a document. *Word segmentation* is defined as the process of extracting words from a given line. *Character segmentation* is defined as the process of extracting the individual characters that constitute the word unit.

Horizontal projection of a document image is most commonly employed to extract the lines from the document. If the lines are well-separated and are not tilted, the horizontal projection will have well-separated peaks and valleys. These valleys are easily detected and used to determine the location of boundaries between lines. This simple strategy fails if the lines are tilted with respect to the horizontal axis of the image plane. The peaks and valleys would not be distinctive for tilted document images, and the text lines cannot be separated by horizontal boundary lines. There are two approaches to handle the problem of tilt. In the first approach the tilt is estimated and the document image is corrected for tilt prior to line segmentation. In the seond approach, tilt correction is applied to each line after line segmentation has been performed.

Vertical projection of a line image is employed in the extraction of words in a line, analogous to the process of line segmentation. Again this simple strategy fails if the words have a slant with respect to the vertical axis of the image plane. To resolve the problem, it is necessary to perform slant estimation and correction. Slant correction is effective and widely used as a preprocessing operation for character segmentation in a word. Slant estimation and correction are also useful in detection and recognition of printed italic letters.

Some documents are handwritten along preprinted or hand-drawn underlines, which interfere with the segmentation and recognition of words and characters. In such cases, it is necessary to detect and eliminate the underlines. A simple underline elimination algorithm is also described in this section.

Tilt Correction

Tilt Correction Before Line Segmentation

Zero Crossing Method (3). Tilt correction is generally performed in two steps. In the first step, the tilt of the document is estimated and in the second step shear transformation is applied to remove the tilt. Tilt or skew is estimated by finding the direction in which the projection of the document has minimum separability with regard to the peaks and valleys in the projection profile. In the zero crossing method, only crossing points are counted to obtain the projection. The use of the zero crossing points rather than the entire foreground pixels is advantageous both in improving the separability of the projection and in saving computation time. The variance of the number of zero crossing points is used as a simple measure of separability. To find the direction which maximizes the separability measure, multiple projections in different directions differing by one or two degrees are calculated within the range of the expected tilt. Once the tilt of the image is estimated, tilt correction is implemented as a shear transformation.

It is worth observing that this straightforward enumerative search for maximum separability is more efficient than expected, if it is implemented properly: All the multiple projections are calculated in a single raster scan. Only the zero crossing points are projected in multiple directions. The mapping is performed incrementally without any multiplication. If the document image is quite large and has sufficient resolution, the raster scan can be performed for every two pixels on alternate scan lines. This interleaving is equivalent to processing a down sampled image, and the process is more efficient, unless the down-sampled image itself is needed in succeeding processes.

During the process of tilt estimation, several other characteristics of the document image (e.g., the number of lines, the interval, and the average height of characters) can be estimated, which are of great use in subsequent operations. Figure 3 shows an example of tilt correction. In this example, tilt is estimated as the direction which maximizes the variance of crossing counts was selected over a range varying from $-8°$ to $+8°$.

Hough Transformation. Hough transformation is one of the most common technques for detection of line segments in an image. It maps the original image to a θ–ρ parameter plane, and a line in the original forms a cluster in the parameter plane. Once the location of the clusters are determined, the

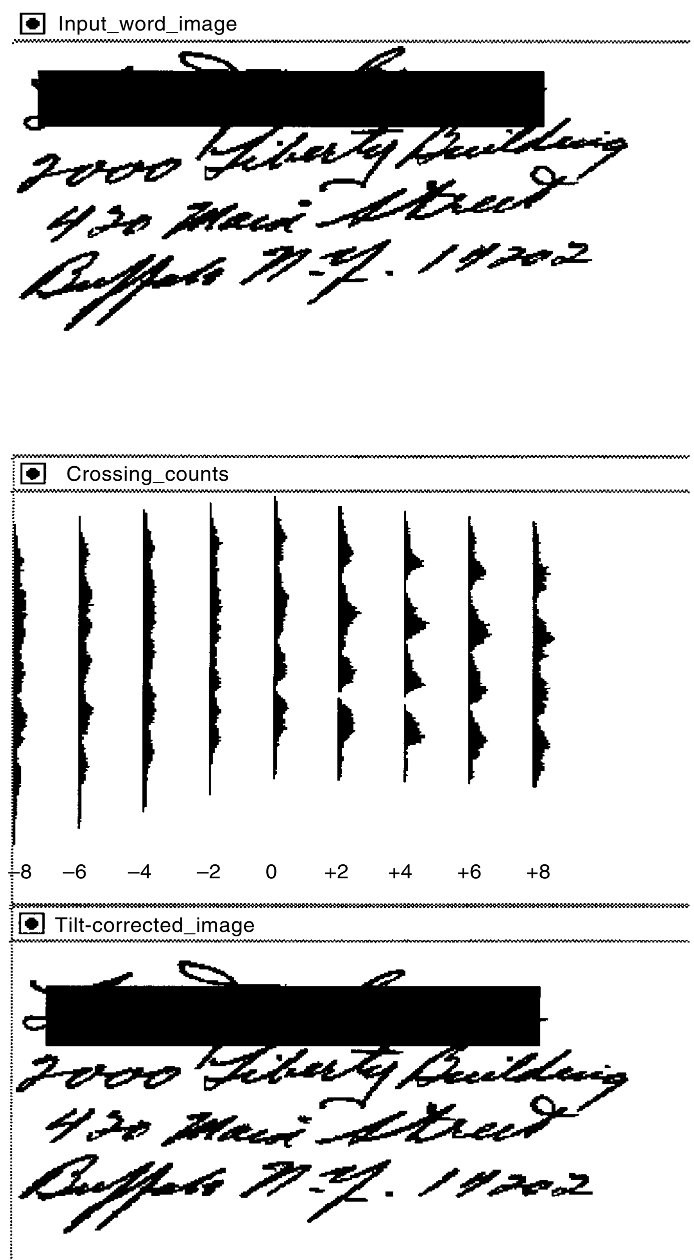

Figure 3. Example of tilt correction.

tilt of each line and the average tilt is easily estimated. The Hough transformation is very attractive and useful because it is available to detect not only solid lines but also broken lines and even text lines consisting of characters and words. This generality, however, sacrifices the processing efficiency; also, the implementation has to be performed carefully, utilizing domain specific knowledge: To reduce the processing time, only border points or crossing points are required to be mapped to the parameter plane instead of the entire foreground pixels. Further reduction is achieved by restricting the range of θ in the parameter plane. It should be noted that if the transformation is performed only for specified θ (e.g., every 2° from $-8°$ to $+8°$), the Hough transformation approach is equivalent to the crossing point method described above. A section of the parameter space at a specified θ is simply the pixel projection in θ direction.

Tilt Correction After Line Segmentation. Techniques for tilt estimation in an entire document such as an address block generally yields an average value; individual lines in the document may still exhibit residual tilt that needs to be removed for more accurate word recognition. A different approach is needed to estimate tilt in a single line of words. It is also observed that the method described in the previous section may yield incorrect tilt estimates, when the number of lines in the document image is less than two. A common approach to estimate tilt in a single-line image is the use of a least-squared error line fit to the bottom profile of the line image (ignoring any descenders in profile derivation). The slope of the resulting line yields a good estimate of the tilt in a single line image.

Underline Elimination

Projection Method. Many techniques have been proposed for the detection and elimination of underlines. These include methods based on Hough transform and morphological operations with suitable structuring elements. An approach based on a line fit to the bottom profile of a single line image has also been used to eliminate underlines characterized by low curvature segments. In a handwritten document consisting of several lines of data, vertical extents of underlines are simply estimated in the horizontal projection of the tilt corrected document image. Within the vertical extents, short vertical runs which are isolated in the extent are removed. A novel algorithm using a morphological approach has been proposed by Liang et al. (4) for removing interference strokes, including underlines from word images. This method is capable of removing hand- and machine-drawn underlines, even when these underlines cut across the characters of the word image.

Line Segmentation

Line Segmentation Before Tilt Correction

Zone Method (5). The algorithm divides an input image into vertical zones. For each zone, a horizontal projection is computed. A vertical extent of the projection with nonzero values form a block. Blocks which are horizontally adjacent and vertically overlapping are connected to form text lines. Heuristics are used to split and/or join blocks. Connected components that are located entirely in blocks from a single line are assigned to that line. Connected components that span more than one text line are split into two or more components and placed in separate text lines.

Morphological Method (6).. In the morphological method, core regions of the image are generated using a morphological operation. These core regions generally fill in the body of each word, but eliminate ascender and descender strokes. For the core images, a technique similar to the zone method is applied. Because of the preceding morphological operation, less heuristics are required to split or merge blocks (to form text lines) than for the zone method. Zones in the morphological method are assumed to have overlapped areas, while zones in the zone method are mutually disjoint.

Line Segmentation After Tilt Correction

Projection Method. As described earlier, the horizontal projection of a document image is calculated, and the valley points are detected and used to determine the location of boundaries between lines. Some valley points may be merged or removed based on heuristics such as the expected interval of lines. If two valley points are closer than a given threshold, they are merged. The advantage of the projection method is its robustness in dealing with documents containing connected lines due to extenders. The disadvantage is the underlying assumption that line boundaries are horizontal.

Component Clustering Method (7). Line segmentation can be considered as a problem of clustering for the connected components. Each connected component is mapped into a two-dimensional space, in terms of its vertical extents ($y_{\min}$, $y_{\max}$) [Fig. 4(c)].

The clusters are detected using typical clustering algorithms such as the K-means clustering (8). The K-means clustering initially requires K clusters or K centers of clusters. These initial values can be obtained in the process of tilt correction. The projection method may be employed to obtain these initial values, if necessary. To suppress the undesirable influence of small noise components or large components including multiple lines connected by extenders, the extended version of the K-means clustering (called the weighted K-means clustering) is known to be useful. In the weighted K-means clustering, the center of a cluster is the weighted mean of the samples. The weight of each connected component is defined so that the closer the height of the component is to the estimated character height, the larger is its weight.

If the number of clusters is not uniquely estimated, but within a range of 6 ± 1, the K-means clustering is applied for all possible values of K. Among the results, clusters which have poor separability are discarded. Clusters that do not satisfy spatial constraints required by valid document lines are also discarded. The remaining clusters yield the number and the position of the lines, and the components are assigned to these clusters (lines). Multiple line components occupying a vertical extent of more than two lines are split at the line boundary [Fig. 4(b)]. The advantage of the component clustering method is its ability to construct complex line boundaries.

Slant Correction

There are two principal approaches for estimating the slant of a word. These include the projection method and the chain code method. A brief description is provided below.

Projection Method. The average slant of characters in a word or in a line is estimated by the analysis of slanted verti-

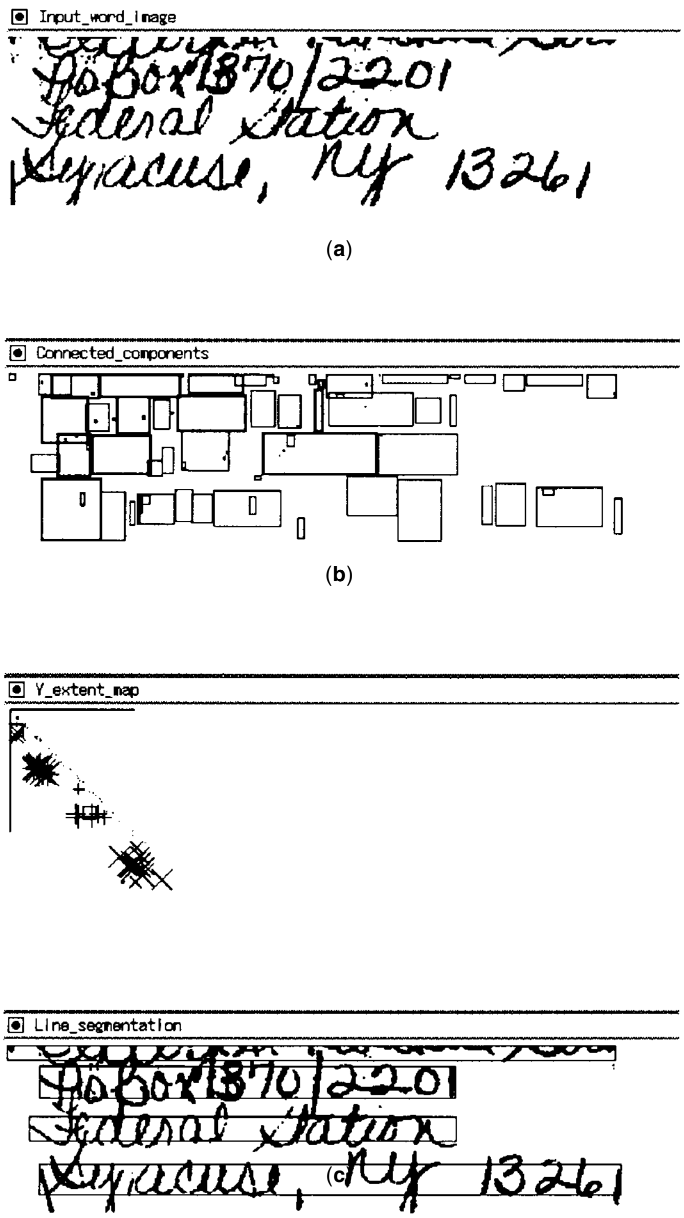

Figure 4. Example of line segmentation. Connected components with thick bounding boxes are subdivided multiline components.

cal projections (histograms) at various angles (9). The average slant is found by looking for the greatest positive derivative in all of the slanted projections.

Chain Code Method (10). In contrast with the tilt estimation, the average slant of characters in a word or in a line is easily estimated using the chain code of the border pixels. The average slope (tangent of the slant angle) is given by

$$m = \frac{n_1 + n_2 + n_3}{n_1 - n_3} \tag{1}$$

where n_i is the number of chain elements at an angle of i times 45° (/ or | or \). Shear transformation is then applied to remove the slant. It is interesting to see why this simple expression gives a good estimate of the average slope. To estimate the slant of characters, only vertical and near vertical edges are useful and horizontal edges only contribute marginally. If horizontal chain elements are removed, the borders of the characters are separated into chain segments having its average slant between 45° to 135°. Since each average slant of these chain segments is calculated by Eq. (1), the overall average is also calculated by Eq. (1). Figure 5 shows an example of slant correction of a word. Figure 5(b) shows the non-horizontal chain elements.

Word Segmentation

Word Boundary Analysis (7). Words are assumed to be separated by a space, a comma, or a period. The space detection algorithm detects the spaces by classifying each gap between the character segments as "between words gap" or "within word gap," respectively. If the gap is wider than a threshold, the gap is classified as "between words gap", otherwise as "within word gap." The threshold, based on the distribution of the gap width for text lines is found by applying a standard technique such as Otsu's method (11).

Exact segmentation of a handwritten line field is very difficult unless it is integrated with the word recognition process. It is interesting to observe that human beings usually employ this approach very efficiently for word recognition. In a typical integrated word segmentation process, the word segmentation is assumed to yield oversegmented word images, where some words can be split and divided into subwords. These (pre-) segments are merge again into a whole word through multiple application of the word recognition algorithm to a set of successive segments. To obtain an oversegmented word image, the line is subject to further segmentation, if undersegmentation is anticipated. If the number of words in a line is too few or the estimated length of a word is too long, the word is divided at the maximum within-field gap. This procedure is repeated until no further subdivision is necessary.

Convex Hull (12). Different metrics can be employed to measure the spatial gaps with varying degrees of accuracy (12). Convex-hull metric requires computation of convex hulls for each of the components in a line. The distance between

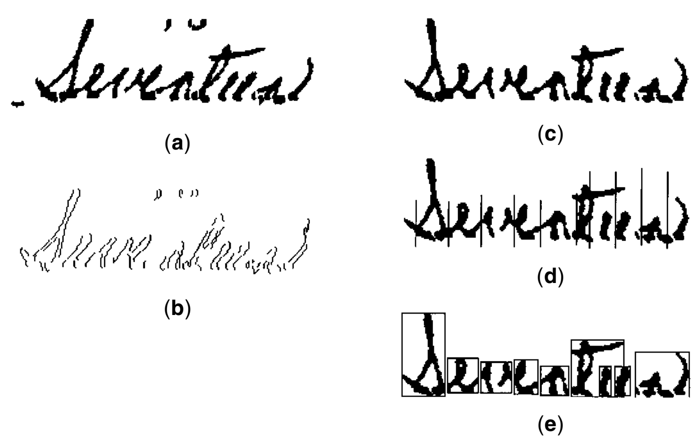

Figure 5. Different stages of processing for word recognition: (a) Original image, (b) extracted chain code, (c) image slant corrected, (d) oversegmented image, (e) final segmentation-recognition.

the convex hulls of the components along the line joining their centers of gravity is used as a distance measure.

RECOGNITION ALGORITHM

Before segmenting a word into its character components, slant estimation and correction are applied to the word image. Segmentation points (character boundaries) are then detected for splitting the word at these segmentation points. As is the case with word segmentation, most character segmentation techniques are designed to generate oversegmented character images. The character segments are merged into a whole character in the succeeding process of character recognition or word recognition. Detection of segmentation points are based on the shape analysis of the word image. Contour analysis, profile analysis, and run-length analysis are most commonly used for this purpose. Figures 5 and 6 illustrate the process of word recognition.

Character Segmentation

Contour Analysis (13, 14). The contour analysis method is suitable to obtain oversegmented character images. Possible segmentation points are detected through local extrema analysis of the upper contour of the word image. Among the local minima, those that are not deep enough from the adjacent local maxima are sequentially removed. In order to obtain characters separated by vertical lines, segmentation points determined in the previous step are often shifted horizontally to the right or the left as follows:

If the minimal point is not open vertically upward, the point is shifted to the right or to the left of this point depending on where the number of runs and the total length of runs is minimal. Figure 7(a) illustrates this process.

Profile Analysis (15). Instead of the upper contour of a word, the upper profile is analyzed. Here the upper profile is defined to be the set of the topmost foreground pixels in each column. Postprocessing is required to find some segmentation points overshadowed by character segments such as the long “t” bar.

Run Length Analysis (16). Ligatures between “or”, “on”, etc., occasionally do not have a valley point in the upper contour. To detect and split these ligatures, a single-run stretch is detected and split at the middle point. The run is vertical streaks of one or more black pixels, and the single-run is the unique run on a single vertical line. The single-run stretch is a horizontal stretch of single-runs shorter than a threshold determined depending on the average stroke width. Figure 5 illustrates this analysis. Among these single-run stretches, those which have well-defined peaks and valley points in the upper contour are removed. The remaining single-run stretches are split at the middle point. Figure 7 shows an example when the ligature connecting “r” and “n” of California had to be cut.

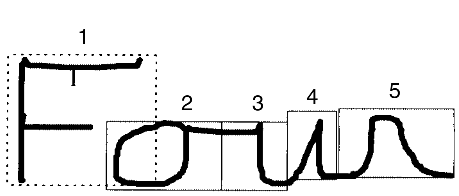

Figure 6. Segmentation-recognition with DP.

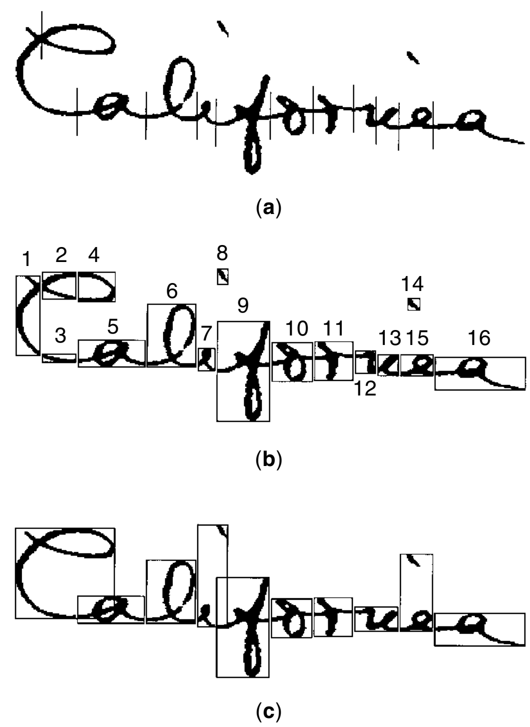

Figure 7. (a) Example of presegmentation points, (b) disjoint box segmentation, and (c) optimum character segmentation.

Disjoint Box Segmentation (13,14). Characters and character segments are more easily handled if their bounding boxes are mutually disjoint. If oversegmentation is permitted, horizontal overlapping of character segments is resolved by a simple algorithm. A word image is split vertically at each presegmentation point and is separated into horizontally nonoverlapping zones. A connected component analysis is applied to the split image to detect the boxes enclosing each connected component. These boxes are usually disjoint and do not include parts of other connected components [Fig. 7(b)]. The disjointedness of the bounding boxes is necessary for high-speed feature extraction.

WORD-MATCHING ALGORITHM

Lexicon-Directed Algorithm (13, 14)

The number of boxes (or segments) obtained by the disjoint box segmentation is generally greater than the number of characters in the word image. In order to merge these segments into characters so that the final character segmentation is optimal, dynamic programming (DP) is applied, using the total likelihood of characters as the objective function. The likelihood of each character is given by a discriminant function. To apply the DP technique, the boxes are sorted left to right according to the location of their centroids. If two or more boxes have the same x coordinates at the centroids, they are sorted top to bottom. Numbers above the boxes in Fig. 7(b) show the order of the sorted boxes. It is worth observing that the disjoint box segmentation and the box sorting process reduce the segmentation problem to a simple Markov process, in most cases. For example, in Fig. 7(b), boxes 1 to 4 correspond to the letter “C” of California, box 5 corresponds to “a”,

Table 1. Table of Likelihood Values

	5	—	—	—	**6.71**	
	4	—	—	**4.87**	4.57	
$j(k) \uparrow$	3	—	3.00	3.25	—	$L(k, j(k))$
	2	1.65	**3.11**	—	—	
	1	**1.90**	—	—	—	
	0					
$k \rightarrow$		1	2	3	4	
Letter		F	o	u	r	

box 6 corresponds to "l", . . ., and so on. This assignment of boxes to letters can be represented as

```
    i  →  1  2  3  4  5  6  7  8  9  10
   Ai     C  a  l  i  f  o  r  n  i  a
 i(i)  →  4  5  6  7  8  9  10 11 13 15 16
```

where i denotes the letter number, and $j(i)$ denotes the number of the last box corresponding to the ith letter. Note that the number of the first box corresponding to the ith letter is $j(i-1)+1$. Given $[j(i),\ i = 1, 2, . . .; n]$ the total likelihood of the character is represented by

$$L = \sum_{i=1}^{n} \ell(A_i, j(i-1)+1, j(i)) \tag{2}$$

where $\ell(A_i, j(i-1)+1, j(i))$ is the likelihood for the ith letter.

In the lexicon-directed algorithm, an ASCII lexicon of possible words is provided and the optimal character segmentation is found for each lexicon word. All lexicon words are then ranked regarding the optimal likelihood per character (L^*/n) to select the best candidate word. The optimal assignment (the optimal segmentation) which maximizes the total likelihood is found by using the dynamic programming technique described below. The optimal assignment $j(n)^*$ for the nth letter is the one such that

$$L^* = L(n, j(n)^*) = \operatorname*{Max}_{j(n)} L(n, j(n)) \tag{3}$$

where $L(k, j(k))$, the maximum likelihood of partial solutions given $j(k)$ for the kth letter, is defined and calculated recursively by

$$L(k, j(k)) = \operatorname*{Max}_{j(1), j(2), \ldots, j(k-1)} \left\{ \sum_{i=1}^{k} \ell(A_i, j(i-1)+1, j(i)) \right\}$$

This can be written in a recursive form as shown below:

$$= \operatorname*{Max}_{j(k-1)} [\ell(A_k, j(k-1)+1, j(k)) + L(k-1, j(k-1))]$$
$$\text{with } L(0, j(0)) = 0,\ j(0) = 1, 2, \ldots, m \tag{4}$$

Starting from Eq. (5), all $L(k, j(k))$'s are calculated for $k = 1, 2, . . ., n$ using Eq. (4) to find $j(n)^*$ using Eq. (3). The rest of $j(k)^*$'s ($k = n-1, n-2, . . ., 1$) are found by back-tracking a pointer array representing the optimal $j(k-1)^*$'s which maximizes $L(k, j(k))$ in Eq. (4).

The example illustrated in Fig. 6, Table 1, and Table 2 shows the values of L and $j(k-1)^*$ for given k and $j(k)$. In this example, $L(n, j(n))^* = L(4, 5) = 6.71$ and $j(n)^* = 5$. Succeeding $j(k)$'s are 4, 3, 2, 0 respectively. The box "0" corresponds to a virtual box standing for the last box of the letter preceding the first letter "C". Figure 6 illustrates the word-matching procedure based on segmentation-recognition with DP.

Table 2. Search for Optimum Segmentation

	5	—	—	—	4	
	4	—	—	3	3	
$j(k) \uparrow$	3	—	2	2	—	$j(k-1)^*$ given
	2	0	1	—	—	k and $j(k)$
	1	0	—	—	—	
	0					
		1	**2**	**3**	**4**	Letter $k \rightarrow$
		1	**2**	**4**	**5**	$j(k)$
Letter		**F**	**o**	**u**	**r**	

Lexicon Free Algorithm

A lexicon free word recognition algorithm is easily obtained from the lexicon-directed algorithm by simple modification. In the lexicon-directed word matching, character likelihood is calculated for a single letter in a specified position of a lexicon word. While in the lexicon free word matching, the total likelihood for an input word is given by

$$L = \sum_{i=1}^{n} \ell(A_i, j(i-1)+1, j(i)) = \sum_{i=1}^{n} \operatorname*{Max}_{A_i}\{\ell(A_i, j(i-1)+1, j(i))\} \tag{5}$$

instead of Eq. (2). The character likelihood for all letters are calculated and the maximum value and the associated letter $A_i{}^*$ are determined. The word matching process is applied only once for an input word, and the recognition result is given by $A^*{}_1A^*{}_2$ . . ., $A^*{}_n$. When the word length n is unknown, an upper bound is estimated and used as the value of n. It is very convenient that the likelihood array $L(k, j(k))$ and the pointer array for a word of length n includes all the entries for shorter words. The lexicon free algorithm is required when no lexicon is available for numeral string recognition. It is also efficient and suitable if the input word is written neatly and segmentation and recognition of characters are relatively accurate.

***K*-Best Path Algorithm (17).** The lexicon free algorithm can be extended so that it generates not only the optimal word interpretation but also the K best interpretations for a specified K. Among the K interpretations, invalid interpretations are removed through lexicon search and the best valid interpretation is selected.

Expression Matching (18,19). Expression matching is the process of matching a character string against words in a lexicon and computing a measure (called "edit distance") indicative of the degree of match between the given string and the words in the lexicon. Spell-checking operations use expression matching to determine alternative words to correct misspelled words. This process uses three operations:

1. Deletion, where certain letters in the string are dropped to obtain the edit distance between the letter string and a lexicon word of lesser length.
2. Insertion, where letters are added to the string to obtain the edit distance between the string and a longer word in the lexicon.
3. Substitution, where one letter in the string is replaced by another letter to obtain the edit distance.

Edit distance is the minimum cost of using the three operations of deletion, insertion, and substitution to match a given letter string to a word in the given lexicon. Dynamic programming is used in deriving this minimum cost.

Final word selection is based on the smallest edit distance between the letter string and the words in the lexicon. The principal advantage of this process is the ease with which the string generated by the recognition algorithm can be matched against lexicon words, even when the length of the string is different from the length of the lexicon word. In word recognition, expression matching is used as a postprocessing operation to determine the best match between the lexicon words and the optimum letter string derived in a context-free recognition mode.

Table 3. Cumulative Correct Recognition Rate of Word Recognition

Rank of Correct	Lexicon Size	Lexicon Size	Lexicon Size
1	98.01%	95.46%	91.49%
2	98.80%	96.70%	91.78%
5	99.60%	97.86%	94.89%

PERFORMANCE EVALUATION—CASE STUDIES

Performance Evaluation of Word Recognition

A total of 2998 word images extracted from the "bha" database (handwritten address block data collected at Buffalo, New York) were used for this test. The test images included city, street, state, personal, and business names, including abbreviated forms. The style of writers ranged from strictly printed to strictly cursive, and from upright to very slanted. The distribution of writing styles reflected the corresponding distribution for the mail stream.

Each word image in the test database was associated with three separate ASCII lexicons of size 10, 100, and 1000, respectively. The lexicons were generated through a random selection process from a statistically significant sample containing city, street, state, and personal and business name words. Each of the three lexicons included the ASCII representation of the associated word image, spelled exactly as it appeared in the image. Thus if the word Illinois was wrongly spelled as Illionis, then the lexicon contained the word Illinois. Table 3 shows the cumulative correct recognition rate.

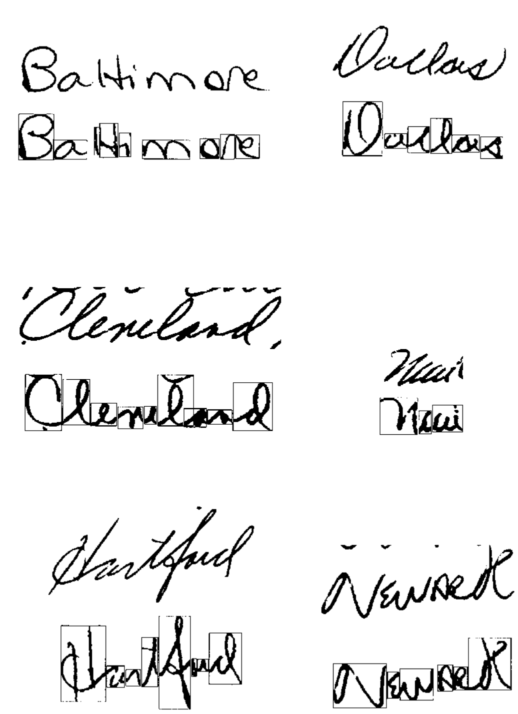

Figure 8. Examples of correctly recognized words.

Table 4. Correct Rates of ZIP Code, Street Number, Street Name, and PO Box Recognition

ZIP Code		Street Number		Street Name		PO Box Number	
Top N	Correct Recognition (%)	Top N	Correct Recognition (%)	Top N	Correct Recognition (%)	Top N	Correct Recognition (%)
1	2910 (82%)	1	2093 (77%)	1	422 (58%)	1	540 (71%)
2	3021 (85%)	2	2119 (78%)	2	434 (60%)	2	556 (73%)
3	3045 (86%)	3	2134 (79%)	3	438 (61%)	3	566 (74%)
4	3056 (86%)	4	2141 (79%)	4	440 (61%)	4	572 (75%)
5	3066 (86%)	5	2148 (79%)	5	441 (61%)	5	575 (76%)
6	3070 (86%)	6	2150 (79%)	6	443 (62%)	6	577 (76%)
Rest	449 (12%)	Rest	551 (20%)	Rest	231 (32%)	Rest	182 (24%)
Reject	21 (0.59%)	Reject	7 (0.26%)	Reject	46 (6%)	Reject	0 (0%)
Total	3540	Total	2708	Total	720	Total	759

The character classifier was designed (trained) using the character samples extracted from state name and city name word images in the "Bd" database. The number of characters used for classifier design was 22606 (435 per character in average) and the correct character recognition rate was about 74.2% for the design samples. The top correct recognition rate was 98.01%, 95.46%, and 91.49% for lexicons of size 10, 100, and 1000, respectively.

Figure 8 shows examples of correctly recognized words. The speed of word recognition was 2.0, 2.5, and 3.5 s/word for each lexicon on a SUN SPARC Station 2. The integrated address interpretation system was designed to determine the nine-digit ZIP code by locating and recognizing the ZIP code, the street number, and/or the PO box fields.

The USPS five-digit city/state/ZIP directory consisting of 100,000 records was used to generate a lexicon of city names for city name recognition. The USPS ZIP+4 address directory consisting of 26 million records was used to generate a lexicon of street names for street name recognition. The performance of the integrated system was evaluated using "bha" test samples. All the samples from bha_6000 to bha_7603 were used for this test.

Tables 4 and 5 summarize the performance at different operating points specified in column 1 of the table. The error rate for one set of operating points was 1.12% with 50.19% encode rate. With a different set of operating points, and error rate of 0.87% with 43.12% encode rate was obtained. In other words the system could be tuned to achieve a specified error rate.

CONCLUSION

The performance of the integrated system developed for the US Postal Service exceeded the performance specifications set by USPS for processing handwritten addresses. The integrated recognition system incorporates several novel features such as (1) tunability for adjusting error-rejection rates and (2) lexicon truncation to achieve low error rate and high processing speed.

A word recognition algorithm using the segmentation-recognition approach is shown to be robust, accurate, and commercially feasible. Context-free recognition is shown to be feasible for numeral string recognition, while a lexicon-directed approach is recommended for word recognition.

In conclusion, it can be stated that handwriting recognition is a feasible technology and can be used with advantage in many commercial applications such as address recognition, forms processing, check processing, and so on. As this article is being concluded, the USPS has announced that the Handwritten Address Interpretation system is currently being deployed in some postal sorting centers on the east coast. Preliminary indications are that the systems are performing satisfactorily. Advances in this field can be directly credited to the USPS, which initiated and supported basic research in this field through grants and contracts to industries and educational institutions.

Table 5. Error Versus Encode Rate

t_{t_1}	t_{t_2}	Encode rate	Error	Correct
20.0	5.0	50.19 (803)	1.12 (9)	98.88 (794)
40.0	7.0	43.12 (690)	0.87 (6)	99.13 (684)

BIBLIOGRAPHY

1. L. S. Frishkoff and L. D. Harmon, Machine Reading of Cursive Script, in C. Cherry (ed.), *Information Processing,* London: Butterworth, 1961, pp. 300–315.
2. L. D. Earnest, Machine Recognition of Cursive Writing, in C. Cherry (ed.), *Information Processing,* London: Butterworth, 1961, pp. 462–466.
3. Y. Ishitani, Document skew detection based on local region complexity, *Proc. 2nd ICDAR,* 1993, pp. 49–52.
4. S. Liang, M. Ahmadi, and M. Shridhar, Segmentation of interference strokes using morphological approach, *Proc. 3rd Int. Conf. Document Anal. Recognition,* Montreal, Canada, 1995, pp. 1042–1046.
5. V. Govindavaju, A. Shekhawat, and S. N. Srihari, Interpretation of handwritten addresses in US mail stream, *Proc. 3rd IWFHR,* 1993, pp. 197–206.
6. M. J. Ganzberger et al., A system for handwritten address interpretation, *Proc. 5th Adv. Technol. Conf.,* 1991, pp. 337–351.
7. F. Kimura, Y. Miyake, and M. Shridhar, Zip code recognition using lexicon free word recognition algorithm, *Proc. 3rd ICDAR,* 1995, pp. 906–910.
8. A. Devijver and J. Kittler, *Pattern Recognition,* London: Prentice-Hall International, 1982, pp. 409–410.

9. D. Guillevic and C. Y. Suen, Cursive script recognition: A sentence level recognition scheme, *Proc. 4th IWFHR,* 1994, pp. 216–223.
10. D. Lee and S .N. Srihari, Handprinted digit recognition: A comparison of algorithms, *Proc. 3rd IWFHR,* 1993, pp. 153–162.
11. L. R. Rabiner and B. H. Juang, An introduction to hidden Markov models, *IEEE Acoust. Speech Signal Process. Mag.,* **30** (1): 4–16, 1986.
12. S N. Srihari, V. Govindaraju, and R. K. Srihari, Handwritten text recognition, *Proc. 4th IWFHR,* 1994, pp. 265–274.
13. F. Kimura, M. Sridhar, and Z. Chen, Improvements of a lexicon directed algorithm for recognition of unconstrained handwritten words, *Proc. 2nd ICDAR,* 1993, pp. 18–22.
14. F. Kimura et al., Context directed handwritten word recognition for postal service applications, *Proc. 5th Adv. Technol. Conf.,* 1992, pp. 199–213.
15. E. Lecolinet and J. Crettez, A grapheme-based segmentation technique for cursive script recognition, *Proc. 1st ICDAR,* 1991, pp. 740–748.
16. R. M. Bozinovic and S. N. Srihari, Off-line cursive script word recognition, *IEEE Trans. Pattern Anal. Mach. Intell.,* **11**: 68–83, 1989.
17. C. R. Nohl, C. J. Burges, and J. I. Ben, Character-based handwritten address word recognition with lexicon, *Proc. 5th Adv. Technol. Conf.,* 1992, pp. 167–180.
18. F. Kimura, M. Shridhar, and N. Narasimhamurthi, Lexicon directed segmentation-recognition procedure for unconstrained handwritten words, *Proc. 3rd Int. Conf. Frontiers Handwriting Recognition,* Buffalo, 1993, pp. 122–131.
19. H. Bunke, A fast algorithm for finding the nearest neighbor of a word in a dictionary, Report of Institut für Informatik und Angewandte Mathematik, Universität Bern, Switzerland, 1993.

Reading List

M. Chen et al., Variable duration Hidden Markov Model and morphological segmentation for handwritten word recognition, *IEEE Trans. Image Process.,* **4**: 1995, 1675–1688.

K. Fukunaga, *Introduction to Statistical Pattern Recognition,* 2nd ed., New York: Academic Press, 1990, pp. 76–78.

N. Gorsky, Off-Line Recognition of Bad Quality Handwritten Words Using Prototypes, in S. Impedovo (ed.), *Fundamentals in Handwriting Recognition,* NATO ASI Series F: Computer and Systems Sciences, Vol. 124, Berlin: Springer-Verlag, 1994, pp. 199–217.

F. Kimura, Y. Miyake, and M. Shridhar, Relationship among quadratic discriminant functions for pattern recognition, *Proc. 4th IWFHR,* 1994, pp. 418–422.

A. Kundu, Yang He, and P. Barl, Recognition of handwritten word: First and second order Hidden Markov Model based approach, *Pattern Recognition,* **22** (3): 283–297, 1989.

M. Leroux, J. C. Salome, and J. Badard, Recognition of cursive script words in a small lexicon, *Proc. 1st Int. Conf. Document Anal. Recognition,* St. Malo, France, 1991, pp. 774–775.

E. Oja, *Subspace Methods of Pattern Recognition,* Oxford, UK: Research Studies Press, 1983.

N. Otsu, A threshold selection method from gray-level histograms, *IEEE Trans. Syst. Man Cybern.,* **SMC-9**: 62–66, 1979.

T. Paquet and Y. Lecourtier, Handwriting recognition: Application on bank cheques, *Proc. 1st Int. Conf. Document Anal. Recognition,* St. Malo, France, 1991, pp. 749–750.

B. Plessis et al., Isolated handwritten word recognition for contextual address reading, *Proc. USPS 5th Adv. Technol. Conf.,* 1992, pp. 579–580.

T. Wakabayashi et al., Accuracy improvement through increased feature size in handwritten numeral recognition, *Syst. Comput. Jpn.,* **26** (8): 35–44, 1995.

S. Watanabe and N. Pakvasa, Subspace method of pattern recognition, *Proc. 1st Int. Joint. Conf. Pattern Recognition,* 1973, pp. 25–32.

M. Shridhar
University of Michigan–Dearborn
Gilles Houle
TRW Enterprise Solutions