Reading and Understanding Handwritten Addresses

Topic: Text/Address Understanding

Authors: Edward Cohen, Jonathan J. Hull, and Sargur N. Srihari

Correspondence Address:

Edward Cohen
SUNY at Buffalo
Department of Computer Science
226 Bell Hall
Buffalo, New York 14260

telephone: (716) 636-3191
fax: (716) 636-3463
Internet: edcohen@cs.buffalo.edu

ABSTRACT

A methodology for *understanding* handwritten addresses is proposed. Address understanding is the process of using multiple information sources to assign a five- or nine-digit ZIP Code to an address block consisting of several text lines. This method uses many diverse pattern recognition and image processing algorithms. To fully process the address image, we must perform thresholding, remove underlining, separate lines of text, segment text lines into words, determine the syntax of the image, and recognize digits, characters, and words.

Our approach emphasizes using contextual information that is available in the address to determine a ZIP Code for a mail-piece. While other research efforts concentrate on simply locating and recognizing the ZIP Code, our system allows the use of recognition information from city name, state name, box number, and ZIP Code to develop a better understanding of the address. This recognition information combined with USPS directory information assists in determining the ZIP Code. Tests of the present system on 508 address images result in a 75.2% accept rate with a 1.6% error rate. This paper describes the algorithms used and suggests improvements for future systems.

Reading and Understanding Handwritten Addresses[1]

Edward Cohen, Jonathan J. Hull,
and Sargur N. Srihari

Department of Computer Science
State University of New York at Buffalo
Buffalo, New York 14260

ABSTRACT

A methodology for *understanding* handwritten addresses is proposed. Address understanding is the process of using multiple information sources to assign a five- or nine-digit ZIP Code to an address block consisting of several text lines. This method uses many diverse pattern recognition and image processing algorithms. To fully process the address image, we must perform thresholding, remove underlining, separate lines of text, segment text lines into words, determine the syntax of the image, and recognize digits, characters, and words.

Our approach emphasizes using contextual information that is available in the address to determine a ZIP Code for a mail-piece. While other research efforts concentrate on simply locating and recognizing the ZIP Code, our system allows the use of recognition information from the city name and state name to develop a better understanding of the address. This recognition information combined with USPS directory information assists in determining the ZIP Code. Tests of the present system on 508 address images result in a 75.2% accept rate with a 1.6% error rate. This paper describes the algorithms used and suggests improvements for future systems.

INDEX TERMS: Character Recognition, Handwriting Recognition, Classifier Systems

I. INTRODUCTION

Determining the destination postal code from handwritten addresses (either hand-printed or cursive) is a problem of great importance to automated mail processing. The United States Postal Service (USPS) must process over ten billion mail pieces with handwritten addresses each year. Present optical character recognition (OCR) systems were designed to read machine print and can recognize only a very small percentage of handwritten addresses. Therefore, a large mail volume must be processed by costly semi-automatic or manual processes.

Handwritten addresses have been automatically sorted in some countries by requiring that the postal code be printed in boxes at known locations on a mailpiece. This is

[1] This work was supported by the United States Postal Service Office of Advanced Technology under Task Order 104230-86M-3990.

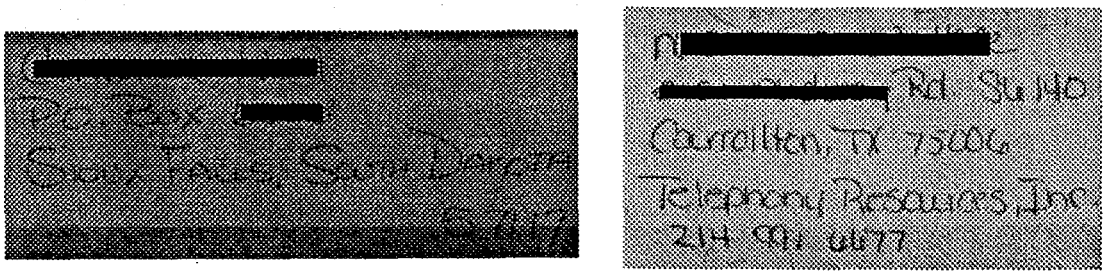
sometimes accompanied by examples of how digits should be printed. This makes the problem much simpler because the postal code is easily located, the digits are separated, and the digits are formed with fewer variations.

Placing stringent requirements on writing addresses has been ruled out in the United States as being publicly unacceptable. Therefore, the algorithms described in this paper were designed to read ZIP Codes and other information in completely unconstrained handwritten addresses.

Handwritten ZIP Code recognition (HZR) is a challenging problem that has several characteristics that distinguish it from the machine-printed OCR problem. These characteristics are illustrated by the examples in Figure 1 that are address blocks taken from live mail. Figure 1(a) is a very neat hand-printed address block in which the ZIP Code digits are prototypical. However, the address contains a "P.O. Box" number which is also a string of digits. Figure 1(b) is a hand-printed address block where the ZIP Code is not in the last line, which makes the ZIP Code location problem non-trivial.

To accurately identify ZIP Codes, other information (such as city and state names) must be located and identified along with the ZIP Code. This "extra" information provides strong evidence about the identity of the ZIP Code. For instance, if the state name was recognized, the first digit of the ZIP Code would be known and the second digit of the ZIP Code could be constrained to (at most) six choices. Our approach in the HZR system (HZRS) is to develop a methodology that enables us to extract information throughout the address.

The complete HZRS needs many components. The input image must be thresholded. Guide lines (horizontal markings used to assist the writer) and noise must be removed. These guide lines tend to connect characters in the image making connected



(a) (b)

Figure 1. Examples of handwritten address blocks extracted from live mail. Addressee names are intentionally blacked out.

component analysis difficult. The text lines must be segmented from the address block, separated into words; and, information word candidates (such as ZIP Code, city name, etc.) must be determined. The information word candidates must be identified. The identification may include segmenting the words into characters (or digits) and using isolated character (or digit) recognition algorithms. Cursive script recognition may also be required.

The system described in this paper contains all the components described above. However, the isolated character and cursive text recognition have only recently been developed; and, therefore, the results on those portions should be considered preliminary. Earlier versions of the system have been described elsewhere [3, 9], but major modifications to the current system have improved performance considerably .

This paper is organized as follows. Section II gives a general discussion of HZRS. Section III details the HZRS algorithms. Section IV covers an example of the algorithms applied to an address. Section V describes performance evaluation and experimental results. Section VI discusses possible refinements to the system.

II. General Discussion of HZRS

Using our approach, we have developed a system that determines the ZIP Codes of handwritten addresses. This system is able to locate and recognize ZIP Codes with a 75.2% accept rate and a 1.6% error rate. A ZIP Code is said to be correctly assigned if a five- or nine-digit ZIP Code is present and all five or nine digits are correctly recognized. The error rate is computed as the ratio of the number of ZIP Codes in error (one or more digits wrong) to the number of images accepted.

HZR is difficult due to wide variations in characters and digits as well as in address structure, but an address has several natural constraints that can be exploited in determining the digits of the ZIP Code. If the state name is known, we can determine the first digit of the ZIP Code and restrict the second digit to at most five choices. If the city name is also known, we can specifically determine the first three digits, and constrain the remaining two digits. Conversely, by knowing the first digit, we can constrain the state name to (at most) seven possible choices. Aside from the redundancy, an address has a spatial syntax that can also be exploited in determining the location of function words (such as city, state, etc.) [7]. The following section describes the details of our current HZRS.

III. HZRS Description

The control flow in the current HZRS has ten operational modules. These modules are listed and described below.

A. Preprocessing;

B. Line segmentation based on positional information;

C. Feature identification;

D. Text line segmentation (combining B & C);

E. Word segmentation;

F. Word classification;

G. Grade match of word classification and syntax;

H. Recognize words that have high syntax-classification match;

I. Use semantics to constrain word recognition;

J. If a reliable semantic evaluation is found, derive an interpretation, otherwise, return to step H until no reliable syntax-classification matches are found;

The following sub-sections A-J give details of each of the ten modules.

A. Preprocessing

Three preprocessing steps are performed on address images in an effort to produce a binary image where text pixels are black and background pixels are white. Two processes (thresholding and underline removal) are unchanged from the previous versions of the HZRS. Interested readers can refer to [3] for details of these two algorithms. The border removal algorithm has not been reported elsewhere and is described below.

Border Removal

An artifact of the digitization process is that some images have a very dark border near the image edge[2]. The borders are detected by scanning the image edges for large black areas (the borders are black after thresholding). Pixels which border both the large black areas and the inner image are used to determine a best fit line along the border. A best fit line is used to compensate for local variations along the edge. After the fit, pixels between the best fit line and the nearest image edge are set to white, effectively removing the border.

B. Initial Text Line Segmentation

Before a proper semantic interpretation can be achieved, the words or digit strings must be located in the image. A line segmentation algorithm separates the image into text lines. The line segmentation is divided into two steps. In the first step, positional information is used to determine line segmentation for "easy-to-place" components. These initial results are refined in step D when feature identification information is available.

The initial line segmentation begins by performing a bottom-up technique we call *shading*. The shading algorithm takes a binary image as input, and divides the image

[2] This was caused by the presence of black construction paper on the digitizing table. If an address was written near the mail-piece edge, the black construction paper might have come into the field of view. When digitized, the construction paper becomes a very noticeable black, i.e., the gray levels are very low.

into vertical strips, (currently 100 pixels wide for our 300 ppi images). For each strip, a horizontal profile (containing one position for each pixel row) is made. Each position in the profile (for each strip) is set to 1 if any image components pass though that strip's row. Otherwise, the position in the profile is set to 0. After a strip's profile is determined, the number of blocks in the strip is determined. A block is a group of adjacent profile positions which have a value of 1. These blocks can be considered portions of the strip which contain image components (see Figure 2). All block positions are recorded, unless they are too small (vertically), in which case the undersized blocks are considered noise and are ignored.

After all blocks from all strips are created, the blocks are connected to form lines (of address block text). Testing has shown that blocks from adjacent strips which overlap vertically are usually from the same address text line. Blocks which are horizontally adjacent and vertically overlapping are connected, forming a topological graph of the blocks, where blocks are represented as nodes and block connections are represented as node connections. Normally, the topological map requires adjustment to reflect the actual address block line. We use heuristics to separate and/or join blocks in each vertical strip and their inter-block connections.

Once the arrangements of blocks into text lines is determined, connected components that are located entirely in blocks from a single line are assigned to that line. Connected components that overlap blocks from several lines are assigned to the set of lines those components overlap. These overlapping connected components are refined in step D.

C. Feature Identification

Feature identification is performed in two stages, shape identification and spatial confirmation. In the first stage, each connected component is identified as having (or

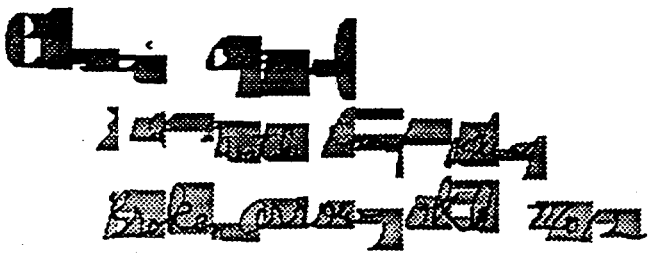


Figure 2. Address block with shaded portions representing the blocks. Each text line in the address has a slightly different color shading, indicating that the shading algorithm determined the line structure correctly.

not having) the proper shape of a feature. Features are currently dashes, commas, disconnected 5-hats[3] , and digits. During this stage, connected components may have multiple feature identifications. For instance, a short horizontal bar may be identified as having the shape of a dash and a disconnected 5-hat.

Spatial and line segmentation information is then used to determine if a component has the proper positional characteristics. For example, to be a disconnected 5-hat, the short horizontal bar must have a left-neighbor component on the same line, the left-neighbor component must not be a comma, dash or disconnected 5-hat, the left-neighbor component must either be identified as a 5 digit or an unrecognized digit, etc.

The constraints for shape identification and spatial orientation are not disjoint. If a component has good positioning, its shape constraints will be relaxed (and vice-versa). For instance, if a component is in a likely "comma location", shape constraints are relaxed to allow odd-shaped commas to be identified as commas.

D. Final Line Segmentation

In some cases, line segmentation cannot be performed based solely on positioning of connected components. Some connected components are positioned closer to another text line than the actual text line to which they belong (e.g., commas are often closer to the text line beneath their actual text line). We have developed a set of heuristics based on empirical studies of handwritten text.

For instance, if a component has the shape of a disconnected 5-hat, and the component is positioned towards the bottom of the text line in which it is currently placed, the component is definitely not a disconnected 5-hat in that line. We then determine the likelihood of this component being a disconnected 5-hat in the text line beneath the current text line. That is, if this component were placed in the text line beneath the current text line, we determine if the component would "fit" as a disconnected 5-hat. If the component "fits" well as a disconnected 5-hat in the lower text line, the component is placed there as part of the line segmentation.

In a similar manner, components discarded as noise can be placed back into text lines. Components with the shape of commas and dashes are sometimes discarded due to their small size. However, if these components have a shape and position that is consistent with a comma, dash, or a disconnected 5-hat, these components can be placed back into the text lines.

In addition to moving components to other text lines, some connected components must be forcibly segmented and placed in more than one line. These components are typically formed by characters from adjacent lines that touch. In these cases, we use connected components with high-confidence line placement to determine where a connected component should be split by forced segmentation. Using those high-confidence connected components, we can determine best fit lines between text lines. That is,

[3] Handwritten 5's are often written in two strokes. The vertical bar and the bottom curve are written in one stroke and the horizontal bar is written as a separate stroke. When the horizontal stroke is not connected to the other stroke, the horizontal stroke is considered to be a disconnected 5-hat.

using the corner points of the bounding boxes around the high-confidence connected components, we determine a dividing line between two text lines.

Although we cannot guarantee perfect performance even after final line segmentation, the results are usually good enough to allow the address to be parsed into words. Furthermore, when the line segmentation process fails, it usually fails in just part of the address (i.e., most of the lines are still segmented properly). Once text line segmentation is complete, the text lines can then be segmented into words.

E. Word Segmentation

To perform word segmentation, each text line is segmented into clusters of connected components, where each cluster can be considered a word. Words are formed by ordering the components of a text line from left to right and selecting word breaks between components (e.g., if components were in order 1-8 and a word break was selected between components 3 and 4, then we would form two words, components 1-3 and components 4-8). Word breaks are hypothesized based on horizontal spaces between connected components, a shift in the mean vertical position of connected components, and the location of commas. The word break locations are ordered based on their confidence (i.e., a large gap would be a high confidence word break). Then, different word break groupings are chosen to separate the text line into 1-5 words.

Note that a text line may be segmented into a given number of words in several ways. Each word break grouping creates different words, and each configuration is assigned a confidence. Each word created by word segmentation is classified as described in the next section.

F. Word Classification

Words are classified based on the set of connected components which comprise that word. The features and the spatial locations of those features determine the word classification. Words categories are *text, city, state-abbreviation, digits (including an estimate of the number of digits), digits-dash, 9-digit ZIP Codes, barcodes, and noise.*

Heuristics are used to classify each word into one or more of these categories and assign a confidence to the word. For example, if a word contained 4 connected components (3 components recognized as digits and the fourth not recognized as a digit or another feature), one word classification would be *digits*. The "digit-size" of the unrecognized component would be estimated based on the ratio of the average size of the 3 recognized digits and the unrecognized component size. If the unrecognized component was 2.2 times as large as the average recognized digit size, the word would be classified as a "5-digits" word (by estimating that the unrecognized component was actually two digits). The word classification confidence would be assigned based on the ratio of the number of recognized digits to the number of connected components in the word.

This word would also be classified as "text", with a lower confidence. The text classification is needed, because text (particularly hand-printed text) can be

misrecognized as digits. It is important that the correct classification is among the classification choices, even if extraneous classifications are made (this allows all consistent line interpretations to be examined). Naturally, too many extraneous classifications would increase the search space and hamper discerning the true classification. In our testing, each word is typically classified into two or three categories.

G. Grading Word Categories to Syntax

Addresses have a syntax that is generally followed. Even when the syntax is not followed precisely, portions of the address will have reasonable syntax. A statistical analysis of information and its location in handwritten mail pieces is found in [4]. Using the statistical analysis and empirical studies of our handwritten address database, we have developed a technique to match word classifications to the most probable word syntax.

The syntax is two-dimensional, since words are horizontally ordered along a text line and the text lines are ordered vertically. One common configuration for handwritten addresses is for the bottom-most line to contain 3 words, city, state, and ZIP Code (CSZ). To determine if the text in a handwritten address contains this configuration, we must see if the bottom-most line can be divided into three words and if each of these three words can be matched to the expected form for each word. Furthermore, we must allow for some inexact matching. The first of the three words must have the characteristics of a "city" word. In the word classification step, this first word may have been classified as a "city", and we would have found a match. However, the word may have been classified as "text" and not "city". Although either of these classifications would be a match to "city", the first classification would be a higher reliability match. A similar approach is required for matching the state name and ZIP Code.

Grading matches between image text and syntax is achieved by ranking high-confidence syntax matches with a highly probable syntax over those with low-confidence syntax matches with less probable syntax. For example, three words classified as "city", "state-abbreviation", and "5-digits" would match to the CSZ syntax better than three words classified as "text", "text", and "4-digits". Note: if the number of digits was estimated, a word classified as having 4 digits might not actually contain 4 digits (e.g., it may contain 5 digits).

When the match grading is complete, we try to locate ZIP Code candidates. The HZRS looks for the top three syntax matches that locate ZIP Code candidates. That is, if a syntax match has the word "ZIP Code" as one of its components, and the syntax match was graded as one of the top three syntax matches containing the word "ZIP Code", then the image text associated with that word would be a ZIP Code candidate. In addition to locating ZIP Code candidates, the syntax matches allow us to find other words (such as the city name, etc.) that (when recognized) can help refine the semantic interpretation of the ZIP Code.

H. Recognizing Words

Once we have determined the location of words, the next step is to recognize important words using techniques designed for those types of words. When a ZIP Code is located, we segment it into digits and recognize the digits. For other types of words, we are developing other word recognition techniques which may not use the traditional segment and classify methodology. The segment and classify methodology is usually most successful for machine-printed text where the boundaries for characters or digits are either well-determined or can be guessed with reasonable accuracy. However, selecting boundaries is more difficult in unconstrained handwritten addresses where the text can be cursive or printed and where the letters or digits can be touching. Our recognition for words composed of characters is currently limited to segmenting and recognizing isolated hand-printed characters. Cursive word recognition is under development.

We now describe the details of ZIP Code recognition. Before its digits are recognized, ZIP Code recognition requires several preprocessing steps. The objective of these steps is to reduce the immense amount of variability present from one ZIP Code to another and to give the digit recognition algorithm images that are as free from writer-dependent effects as possible. These segmentation, preprocessing, and recognition algorithms are described below.

ZIP Code Segmentation

Given an image which contains a full string of ZIP Code digits, the ZIP Code segmentation algorithm will segment the whole image into regions each containing an isolated digit (see Figure 3). The resulting images from this subsystem will be passed to an isolated digit recognition subsystem. In general, a legal ZIP Code consists of either five or nine digits with or without a dash in the middle, the ZIP Code segmentation algorithm applies this information to invoke its splitting or merging operations. The

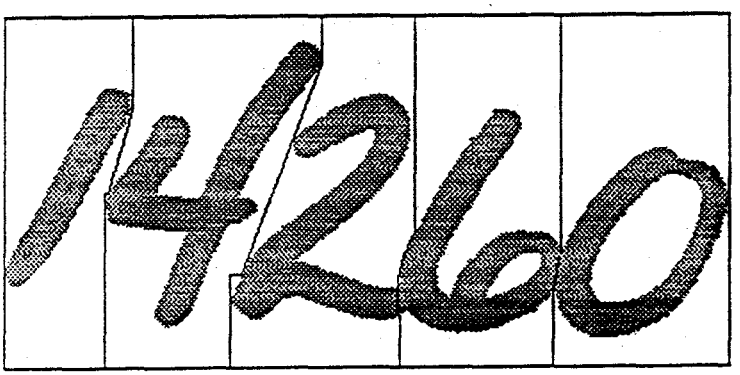


Figure 3. An example of ZIP Code segmentation.

technique consists of the following major phases: connected component analysis, estimation of the number of digits in the input image, and grouping and dissection.

First, the physical attributes (location, height, width, area size, etc.) of each connected component in the ZIP Code image are recorded. Other attributes based on the whole string of digits in the image are also computed and recorded. These attributes consist of its upper and lower contour profiles, the topological relationship between connected components (e.g., left of, right of, etc.), and the positions of ZIP Code string's upper line, central line, and lower line. In addition to the creation of data structures, several operations are performed on the image (e.g., sorting connected components according to their heights, selecting the digit candidates from all the connected components, detecting and removing of underlining, removing noise or insignificant components, removing long ligatures on the tails of digit candidates, etc.).

Using these attributes, the program estimates the number of digits in the image. In addition, the spacing between connected components is used to group components into digit clusters, where each cluster contains one or more digits. We also estimate the number of digits in each cluster.

Grouping and dissecting operations are performed as needed to generate isolated digit images from the clusters. To extract non-touching digits, line segments are drawn to enclose zones that contain isolated digits. Grouping operations merge separate connected components and their containment areas into a single digit zone. Grouping is required when a digit is fragmented due to thresholding or when more than one connected component constitutes a complete digit (digits 4 and 5 are the most frequently seen cases). The grouping function includes the merging of connected components in either neighboring clusters or a single cluster. Dissecting operations split connected digits into isolated ones, and try to maintain the original digit shapes for recognition purposes. Once the number of touching digits is determined, the splitting routine selects the dissecting positions and dissects the connected component into that number of digits. Several factors influence the selection of dissecting positions: the slant angle of the whole ZIP Code string, the average width of digits in the image, the peaks and valleys detected from the lower and upper contour profile respectively, and the number of strokes near where touching is predicted. Based on factors found, the dissector will use different techniques to separate digits. For example, if both a peak and a valley are found near the predicted position, the dissector will segment the gap between two digits by directed cutting through the image from the peak to the valley. If only a peak or valley is found, a similar operation called "hit and deflect" is performed. This technique is always guided by the slant angle. If no peaks or valleys are found, a cut is made through a "single stroke" area.

After all the proper actions have been taken, each enclosed single digit zone is extracted and output with a size of its minimum bounding rectangle.

Digit Recognition

Several algorithms for handwritten digit recognition have used a hierarchical approach in which more than one basic algorithm is applied either in sequence or in parallel. This is done to achieve (i) higher speed by applying the most efficient algorithms first and accepting their results only if confidence is high enough and (ii) greater accuracy by combining the results of more than one algorithm [1,2]. We have followed a parallel approach in order to achieve a high recognition rate and a low error rate. Our method includes four algorithms: (i) a polynomial discriminant algorithm to make a decision based on the overall holistic characteristics of the image[8], (ii) a mixed statistical and structural classifier used on features extracted from the contour of the character [3], (iii) a structural classifier used on information about the size and placement of strokes in the image [6], and (iv) a rule-based algorithm based on chain-code features [5]. These four algorithms were chosen because they utilize different types of information in the image and thus have a better chance of compensating for each other's weaknesses. Each of the algorithms is applied to the same digit image (although the polynomial discriminant and the statistical and structural character recognition algorithms require that the digit image is size normalized before recognition) and their results are combined with a decision tree to classify the image.

Combination of Results

The results of applying the four digit recognition algorithms discussed above are combined with the decision tree. The first test in the tree is: If the distance of the polynomial discriminant method is greater than a threshold and the best decision of the four techniques agree then output that decision. Subsequent tests check other combinations of decisions. There are seventeen such tests in the tree.

If the error rate is "normalized" so that it is below 1.5 percent, the best digit recognition performance is 95.02 percent correct with a 1.33 percent error rate. These represent the results of all digits from a set of 497 ZIP Codes where each ZIP Code was automatically segmented into digits.

I. Using Semantic Information to Constrain Word Recognition

Once all relevant words have passed through the initial recognition phase, semantic information is used to further improve recognition performance. We have just begun developing handwritten word recognition algorithms, therefore, we have not fully implemented all our techniques for using semantic information to read address information. Still, preliminary results show that semantic information can be applied in a useful manner.

One straight-forward semantic check is to determine if the recognized ZIP Code is in a USPS directory of all valid ZIP Codes. The US Postal Service has assigned approximately 40% of all possible 5-digit ZIP Codes. Therefore, if errors are generated randomly in the first five digits, it is expected that 60% of the errors will be detected (and removed) by this test.

We also compare the recognized state name to the first two digits of the ZIP Code. If the state name is recognized, several actions can be taken. When the state name and ZIP Code agree, we accept the ZIP Code as correct. If the state name and ZIP Code digits do not agree, one of three actions is taken. If the state name is recognized with high confidence and the first and/or second digit of the ZIP Code has a low-confidence recognition (or the digit is rejected), the ZIP Code will be corrected to correspond to the state name if possible. For example, if the state name is recognized as Illinois and the ZIP Code is recognized as ?0680 (where ? is a rejected digit or low confidence digit), we can determine that the ZIP Code must be 60680. This is because all ZIP Codes from Illinois begin with either 60, 61, or 62. However, if the state name is recognized as Illinois and the ZIP Code is recognized as 6?680, we cannot determine the ZIP Code precisely and the mail piece is rejected. Finally, if the state name is recognized with high confidence as Illinois and the ZIP Code is recognized 14215 with high confidence, the mailpiece is also rejected. We currently have no technique of determining which of the high confidence recognitions is incorrect. As the HZRS evolves and we can confidently recognize other parts of the address image, we hope to use this extra information to resolve ambiguities.

With the development of our handwritten word recognition algorithms, we are poised to take advantage of other semantic information as it becomes available. Even if the state name is not entirely recognized, it can still provide information to assist in determining the ZIP Code. For instance, if only the first letter of the state name is recognized, the number of possible state names can be reduced from 50 to at most 8 (with an average of 2.6). Similar reductions can be gained by recognizing all or part of the city name. If no ZIP Code is present (or it is unrecognizable), the ZIP Code can usually be determined if the city name and state name are recognized.

IV. Example of HZRS on an Address Image

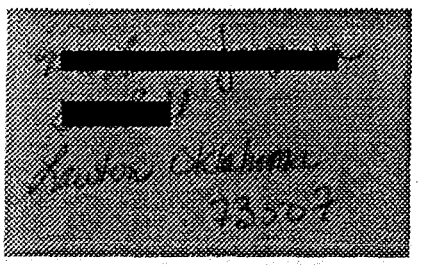
Figure 4 shows some intermediate processing steps of the HZRS. This example demonstrates how state name information can be combined with a poorly recognized ZIP Code to produce the proper ZIP Code result.

The grey level image is shown in (a), and the thresholded image in (b). The personal name and street address have been intentionally blacked out in accordance with our agreement with the USPS. No underline removal is required for this image, but connected component analysis is performed to segment the address into text lines and words.

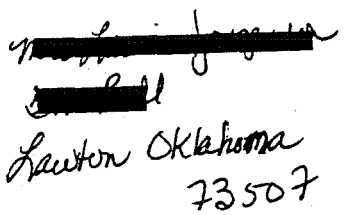
Each text line is segmented into several different word groupings and the selected word segmentation results in a syntax of *text-text-5digits* (c) corresponding to *city-state-ZIP*. Based on this match, we select the 5-digit word as the ZIP Code candidate. The ZIP Code candidate is segmented (d) and the isolated digits are passed to the digit recognition algorithm. The results from digit recognition and their confidences are shown in (e). The recognized ZIP Code *23507* is checked against the USPS ZIP Code directory and found to be valid. However, the low confidence of the first digit (0.735)

would cause the ZIP Code to be rejected.

The state name is recognized as OKLAHOMA which only has ZIP Codes beginning with 73 and 74. Since the state name has high confidence recognition and the ZIP Code has a low-confidence first digit, the ZIP Code is changed to *73507* (f). In this example, the results of the state name recognition, ZIP Code recognition, and USPS directory are combined to produce the proper ZIP Code for the mailpiece.



(a)



(b)

Lawton Oklahoma 73507

(c)

(d)

2 0.735 0.265
3 0.999 0.001
5 0.977 0.023
0 0.999 0.001
7 0.820 0.250

(e)

23507 --> 73507

(f)

Figure 4. Example of HZRS processing an image. (a) Grey-level image. (b) Thresholded image. (c) City, State, and ZIP Code Candidates. (d) Segmented ZIP Code. (e) Digits recognition results for ZIP Code (digit result, correct confidence, reject confidence). (f) Conversion of ZIP Code when recognized state name is used.

V. Performance Evaluation

The current system performance as of 7/23/90 is 75.2% accept rate with a 1.6% error rate. The error rate is the number of improperly accepted ZIP Codes divided by the total number of ZIP Codes accepted. HZRS research is ongoing and system performance is being improved on a (nearly) day-to-day basis. These results are from a set of 508 images chosen by the USPS from the Postal Address Image Database.

For each image, the HZRS system returned one of two responses: a ZIP Code (5 or 9-digit) or a rejection message. A rejection message indicated that the system did not find sufficient information to determine a ZIP Code value. The contents of each image was manually examined and entered into an index. A correct response would be the proper ZIP Code (if present) or a rejection (if no ZIP Code was present). In addition to those ZIP Codes accepted by the system, the system correctly rejected all 11 images (2.2%) that did not contain ZIP Codes.

VI. Discussion

We have shown a system that determines ZIP Codes from handwritten postal addresses. On a test set of 508 address images, the system has a 75.2% accept rate with a 1.6% error rate. The system is evolving as efforts to improve its performance are continuing. We are actively developing cursive word recognition algorithms and hope to add these to our HZRS. Cursive word recognition will allow us to use more of the semantic information available in the addresses.

Acknowledgment

The authors are deeply indebted to Carl O'Connor and Timothy Barnum of the Office of Advanced Technology of the United States Postal Service and Dr. John Tan of Arthur D. Little Inc. for their guidance and support of this work. In addition, we would like to thank Alan Commike, Peter Cullen, John Favata, Chih-Chau Kuan, and Dar-Shyang Lee for contributing ideas and code towards the HZR project.

References

1. P. Ahmed and C. Y. Suen, "Computer recognition of totally unconstrained handwritten ZIP codes," *International Journal of Pattern Recognition and Artificial Intelligence 1*, 1 (Mar 1987), 1-15.
2. B. Duerr, W. Haettich, H. Tropf and G. Winkler, "A combination of statistical and syntactical pattern recognition applied to classification of unconstrained handwritten numerals," *Pattern Recognition 12*, 3 (1980), 189-199.
3. J. J. Hull, S. N. Srihari, E. Cohen, C. L. Kuan, P. Cullen and P. Palumbo, "A Blackboard-Based Approach to Handwritten ZIP Code Recognition," *USPS Advanced Technology Conference* , May 1988 , 1018-1032.
4. J. J. Hull, D. Lee and S. N. Srihari, "Characteristics of handwritten mail addresses: A statistical study for developing an automatic ZIP code recognition system," Technical Report 88-06, Department of Computer Science, State University of New York at Buffalo, Mar 1988.
5. J. J. Hull, A. Commike and T. Ho, "Multiple Algorithms for Handwritten Character Recognition," *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, Montreal, Apr 2-3, 1990, 117-124.
6. C. Kuan and S. N. Srihari, "A stroke-based approach to handwritten numeral recognition," *Proceedings of USPS Advanced Technology Conference*, Washington D.C., May 3-5, 1988, 1033-1041.
7. P. W. Palumbo and S. N. Srihari, "Text parsing using spatial information for recognizing addresses in mail pieces," *International Conference on Pattern Recognition*, Paris, France, Oct 1986, 1068-1070.
8. J. Schurmann, "A Multifont Word Recognition System for Postal Address Reading," *IEEE Transactions on Computers C-27*, 8 (Aug 1978), 721-732.
9. S. N. Srihari, E. Cohen, J. J. Hull and L. Kuan, "A System to Locate and Recognize ZIP Codes in Handwritten Addresses," *International Journal of Research and Engineering Postal Applications 1* (1989), 37-45.