K. Y. Wong
R. G. Casey
F. M. Wahl

Document Analysis System

This paper outlines the requirements and components for a proposed Document Analysis System, which assists a user in encoding printed documents for computer processing. Several critical functions have been investigated and the technical approaches are discussed. The first is the segmentation and classification of digitized printed documents into regions of text and images. A nonlinear, run-length smoothing algorithm has been used for this purpose. By using the regular features of text lines, a linear adaptive classification scheme discriminates text regions from others. The second technique studied is an adaptive approach to the recognition of the hundreds of font styles and sizes that can occur on printed documents. A preclassifier is constructed during the input process and used to speed up a well-known pattern-matching method for clustering characters from an arbitrary print source into a small sample of prototypes. Experimental results are included.

Introduction

The decreasing cost of hardware will eventually make commonplace the storage and distribution of documents by electronic means. However, today most documents are being saved, distributed, and presented on paper. Paper is the primary medium for books, journals, newspapers, magazines, and business correspondence. This paper describes an experimental Document Analysis System, which assists a user in the encoding of printed documents for computer processing.

A document analysis system could be applied to extract information from printed documents to create data bases (*e.g.*, patents, court decisions). Such systems could also create computer document files from papers submitted to journals and assist in revising books and manuals. In addition, a document analysis system could provide a general data compression facility, since encoding optically scanned text from a bit-map into symbol codes greatly reduces the cost of storing or transmitting the information.

Existing commercial systems cannot adequately convert unconstrained printed documents into a format suitable for computer processing. Data entry by operator keying is not only expensive and time-consuming but also precludes the encoding of graphics and images contained in the document. Current optical character recognition (OCR) machines only recognize certain preprogrammed fonts; they do not accept documents containing a mixture of text and images. The Document Analysis System thus provides a new capability for converting information stored on printed documents, including both images and text, to computer-processable form.

This paper is organized into three sections. The first section gives an overview of the system requirements. Not all the components described in the system have been designed and tested yet. The next two sections present the results of the technical investigations of some of the key components: the segmentation of a scanned document into regions of text and images, and the recognition of text. Initial experimental data are included to illustrate these two methods.

System requirements and overall structure

The design of an interactive computer system to read printed documents was investigated by Nagy [1]. Some of the system concepts used by him are reflected in the Document Analysis System, but the solutions in most areas are different.

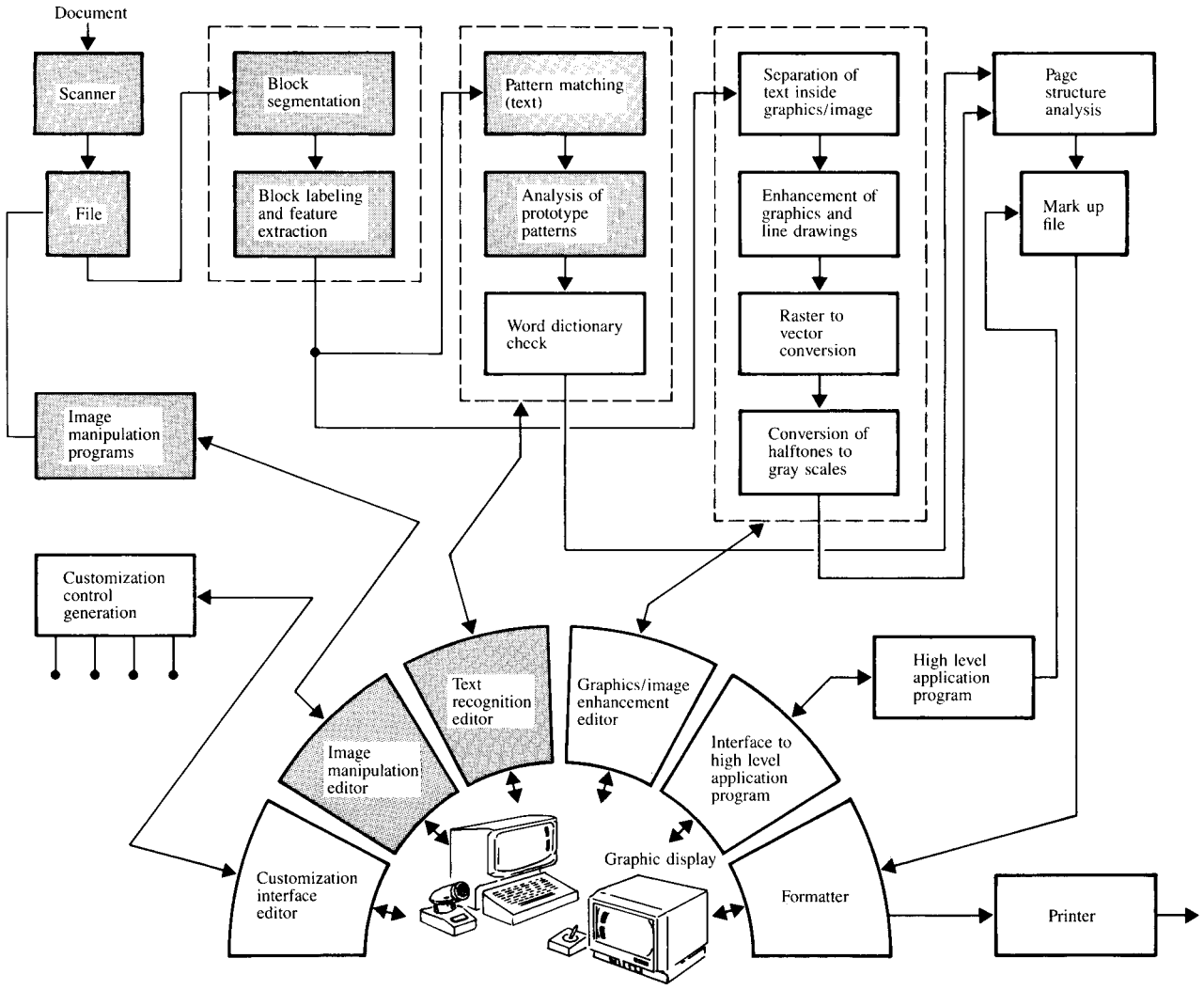

Figure 1 System overview.

The system is designed to have the following capabilities:

- Manual editing of scanned documents (*e.g.*, functions such as crop, copy, merge, erase, save, etc.).
- Separating a document into regions of text and images automatically.
- Representing text by standard symbols such as EBCDIC, ASCII, etc., possibly with the assistance of user interaction. Automatic character recognition will be provided for documents printed in certain conventional fonts.
- Checking the recognized text against a dictionary.
- Enhancing the images and converting graphics from bitmaps into vector format.
- Analyzing the page layout so that appropriate typographical commands can be generated automatically or interactively to reproduce the document with printer subsystems.
- Allowing the user to customize the system for his particular needs.

Figure 1 shows the modules of the proposed Document Analysis System. The shaded modules have been implemented or investigated in detail. The preliminary version of the system permits meaningful experiments but does not yet provide full interactive capability.

Documents are scanned as black/white images with a commercial laser scanner at 240 pixels per inch into a Series/1 minicomputer, which then transfers the data over a network to the host IBM 3033 computer. The display device is a Tektronics 618 storage tube attached to an IBM 3277 terminal, which communicates with the host. The system can automatically separate a document into regions of text, graphics, and halftone images. The technique for doing this is presented in the next section. Alternatively, the user can choose to perform the same function manually by interactively editing the image. The user can invoke the Image Manipulation Editor (called IEDIT), which reads and stores

images on disk and provides a variety of image-editing functions (*e.g.*, copy, move, erase, magnify, reduce, rotate, scroll, extract subimage, and save).

The blocks containing text can be further analyzed with a pattern-matching program that groups similar symbols from the document and creates a prototype pattern to represent each group. The prototype patterns are then identified either manually using an interactive display or by automatic recognition logic if this is available for the fonts used in the document. Inclusion of a dictionary-checking technique is planned to help correct errors incurred during recognition.

Other functions will be provided in future extensions to the system. For instance, the graphics and halftone images separated from the document may need further processing in certain applications. Thus, in a maintenance manual the text labels may need revision without changing the line drawing itself, and vice versa. In such cases, text information must be extracted from the image. Also, due to spatial digitization errors, scanned line drawings and graphics generally have missing or extra dots along edges. Consequently, after scanning and thresholding, identical lines may vary in width in the binary bit-maps. Such errors can be corrected with an image enhancement function. Scanned halftone images may also have moiré patterns [2]. Additional processing functions are required to remove the moiré patterns and convert the images into gray scale format so that conventional digital halftoning techniques can be used to reproduce the image. In another application, existing line drawings and graphics may have to be encoded into an engineering data base. Here, conversion of the drawing from bit-maps into vector or polynomial representation of objects is necessary.

In order to produce the proper sequential list for the text blocks, a layout analysis is required. For example, the text blocks would be linked differently for a single-column than for a multiple-column text page. If the document is to be reproduced exactly, control codes for indenting, paragraphing, and line spacing must be supplied. Finally, typesetting commands may be required to reproduce the document with a printer subsystem.

Block segmentation and text discrimination

Some earlier approaches to segmentation and discrimination [3, 4] required knowledge of the character size in order to separate a document into text and nontext areas. The method developed for the Document Analysis System consists of two steps. First, a segmentation procedure subdivides the area of a document into regions (blocks), each of which should contain only one type of data (text, graphic, halftone image, etc.). Next, some basic features of these blocks are calculated. A linear classifier which adapts itself to varying character heights discriminates between text and images. Uncertain classifications are resolved by means of an additional classification process using more powerful, complex feature information. This method is outlined below; for a more detailed description see [5].

Figure 2(a) shows an example of a document comprised of text, graphics, halftone images, and solid black lines. A run-length smoothing algorithm (RLSA) had been used earlier by one of the authors [6, 7] to detect long vertical and horizontal white lines. This algorithm has been extended to obtain a bit-map of white and black areas representing blocks containing the various types of data [see Fig. 2(d)].

The basic RLSA is applied to a binary sequence in which white pixels are represented by 0's and black pixels by 1's. The algorithm transforms a binary sequence x into an output sequence y according to the following rules:

1. 0's in x are changed to 1's in y if the number of adjacent 0's is less than or equal to a predefined limit C.
2. 1's in x are unchanged in y.

For example, with $C = 4$ the sequence x is mapped into y as follows:

x: 0 0 0 1 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0

y: 1 1 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 1

When applied to pattern arrays, the RLSA has the effect of linking together neighboring black areas that are separated by less than C pixels. With an appropriate choice of C, the linked areas will be regions of a common data type. The degree of linkage depends on C, the distribution of white and black in the document, and the scanning resolution. (The Document Analysis System scans at 240 pixels per inch.)

The RLSA is applied row-by-row as well as column-by-column to a document, yielding two distinct bit-maps. Because spacings of document components tend to differ horizontally and vertically, different values of C are used for row ($C_h = 300$) and column ($C_v = 500$) processing. Figures 2(b) and 2(c) show the results of applying the RLSA in the horizontal and in the vertical directions of Fig. 2(a). The two bit-maps are then combined in a logical AND operation. Additional horizontal smoothing using the RLSA ($C_s = 30$) produces the final segmentation result illustrated in Fig. 2(d). Reference [8] describes in more detail an efficient two-pass algorithm for the implementation of the RLSA in two dimensions.

The blocks shown in Fig. 2(d) must next be located and classified according to content. To provide identification for subsequent processing, a unique label is assigned to each block. This is done by a labeling technique described in [9]. Simultaneously with block labeling, the following measurements are taken:

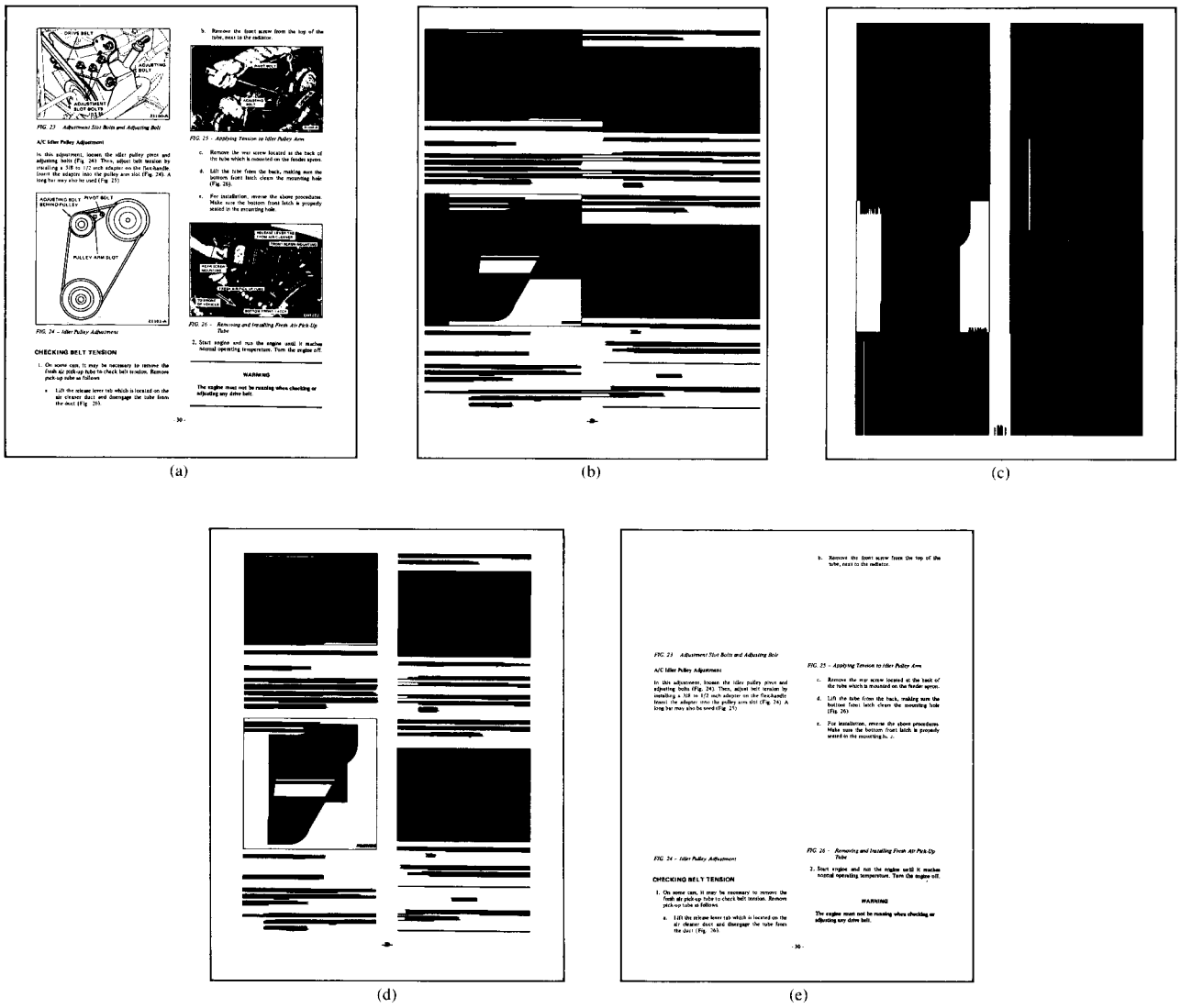

Figure 2 (a) Block segmentation example of a mixed text/image document, which here is the original digitized document. (b) and (c) Results of applying the RLSA in the horizontal and vertical directions. (d) Final result of block segmentation. (e) Results for blocks considered to be text data (class 1).

- Total number of black pixels in a segmented block (BC).
- Minimum x-y coordinates of a block and its x, y lengths ($x_{\min}$, Δx, $y_{\min}$, Δy).
- Total number of black pixels in original data from the block (DC).
- Horizontal white-black transitions of original data (TC).

These measurements are stored in a table (see Table 1) and are used to calculate the following features:

1. The height of each block segment: $H = \Delta y$.
2. The eccentricity of the rectangle surrounding the block: $E = \Delta x/\Delta y$.
3. The ratio of the number of block pixels to the area of the surrounding rectangle: $S = BC/(\Delta x \Delta y)$. If S is close to one, the block segment is approximately rectangular.
4. The mean horizontal length of the black runs of the original data from each block: $R_m = DC/TC$.

These features are used to classify the block. Because text is the predominating data type in a typical office document and text lines are basically textured stripes of approximately a constant height H and mean length of black runs R_m, text blocks tend to cluster with respect to these features. This is

illustrated by plotting block segments in the R-H plane (see Fig. 3). Each table entry is equal to the number of block segments in the corresponding range of R and H. Thus, such a plot can be considered as a two-dimensional histogram. The text lines of the document shown in Fig. 2(a) form a clustered population within the range $20 < H < 35$ and $2 < R < 8$. The two solid black lines in the lower right part of the original document have high R and low H values in the R-H plane, whereas the graphic and halftone images have high values of H. Note that the scale used in Fig. 3 is highly nonlinear.

The mean value of block height H_m and the block mean black pixel run length R_m for the text cluster may vary for different types of documents, depending on character size and font. Furthermore, the text cluster's standard deviations $\sigma(H_m)$ and $\sigma(R_m)$ may also vary depending on whether a document is in a single font or multiple fonts and character sizes. To permit self-adjustment of the decision boundaries for text discrimination, estimates are calculated for the mean values H_m and R_m of blocks from a tightly defined text region of the R-H plane. Additional heuristic rules are applied to confirm that each such block is likely to be text before it is included in the cluster. The members of the cluster are then used to estimate new bounds on the features to detect additional text blocks [5]. Finally, a variable, linear, separable classification scheme assigns the following four classes to the blocks:

Class 1 Text:
$R < C_{21} R_m$ and
$H < C_{22} H_m$.

Class 2 Horizontal solid black lines:
$R > C_{21} R_m$ and
$H < C_{22} H_m$.

Class 3 Graphic and halftone images:
$E > 1/C_{23}$ and
$H > C_{22} H_m$.

Class 4 Vertical solid black lines:
$E < 1/C_{23}$ and
$H > C_{22} H_m$.

Values have been assigned to the parameters based on several training documents. With $C_{21} = 3$, $C_{22} = 3$, and $C_{23} = 5$, the outlined method has been tested on a number of test documents with satisfactory performance. Figure 2(e) shows the result of the blocks which are considered text data (class 1) of the original document in Fig. 2(a).

There are certain limitations to the block segmentation and text discrimination method described so far. On some documents, text lines are linked together by the block segmentation algorithm due to small line-to-line spacing, and thus are assigned to class 3. A line of characters exceeding 3 times H_m (with $C_{22} = 3$), such as a title or heading in a document, is assigned to class 3. An isolated line of text printed vertically, *e.g.*, a text description of a diagram along the vertical axis, may be classified as a number of small text blocks, or else as a class 3 block.

Table 1 Processing result of document in Fig. 2. Columns 1 to 7 contain the results of the measurements performed simultaneously with labeling. The last column is the text classification result.

BC	x_{min}	Δx	y_{min}	Δy	DC	TC	*Class*
702	995	68	2341	23	302	76	1
6089	1090	771	2266	13	5608	170	2
6387	307	265	2245	32	1142	396	1
15396	307	657	2208	31	3118	1005	1
9341	1090	366	2184	31	2469	580	1
16706	185	779	2171	24	3489	1070	1
19447	1090	771	2147	35	5181	1110	1
9244	185	385	2098	31	1780	528	1
3244	1401	152	2077	23	1587	303	1
18502	185	779	2060	32	4112	1183	1
17592	185	779	2024	30	3613	1018	1
5667	1091	770	2008	13	5394	72	2
12300	185	474	1947	28	4751	735	1
19318	1144	717	1923	35	4436	1155	1
19252	1101	760	1887	32	3981	1059	1
11101	188	483	1830	29	2713	756	1
1258	1144	171	1821	22	434	123	1
20200	1082	779	1782	34	4349	1229	1
276147	188	780	1037	766	47742	8738	3
418268	1084	780	1209	546	195328	59906	3
10935	1088	503	1117	30	2139	648	1
18370	1088	773	1080	31	3406	996	1
19120	1088	773	1043	32	3632	1046	1
3434	1205	126	969	32	770	226	1
13694	193	489	954	30	2251	758	1
17336	1088	773	934	31	3417	1012	1
21586	193	783	917	31	3574	1164	1
17601	1088	773	897	32	3580	1040	1
19174	193	779	880	31	3437	1165	1
23306	193	778	840	31	4167	1256	1
15500	1088	773	824	32	3544	1038	1
20263	193	778	804	30	3800	1183	1
16619	1088	773	788	25	3310	989	1
10112	193	393	731	31	2911	666	1
16777	1088	773	705	31	3749	1083	1
385911	1087	778	175	504	175114	51527	3
19512	193	777	643	29	4046	1205	1
406563	193	777	75	538	63103	12598	3
10467	1089	477	115	28	1835	542	1
18717	1089	776	78	30	3427	968	1

Consequently, in order to further classify different data types within class 3, a second discrimination based on shape factors [10] is used. The method uses measurements of border-to-border distance within an arbitrary pattern. These measurements can be used to calculate meaningful features like the "line-shapeness" or "compactness" of objects within binary images. While the calculations are complex and time-consuming, they are done only for class 3, and thus add only a small increment to the overall processing time. This hierarchical decision procedure, in which "easy" class

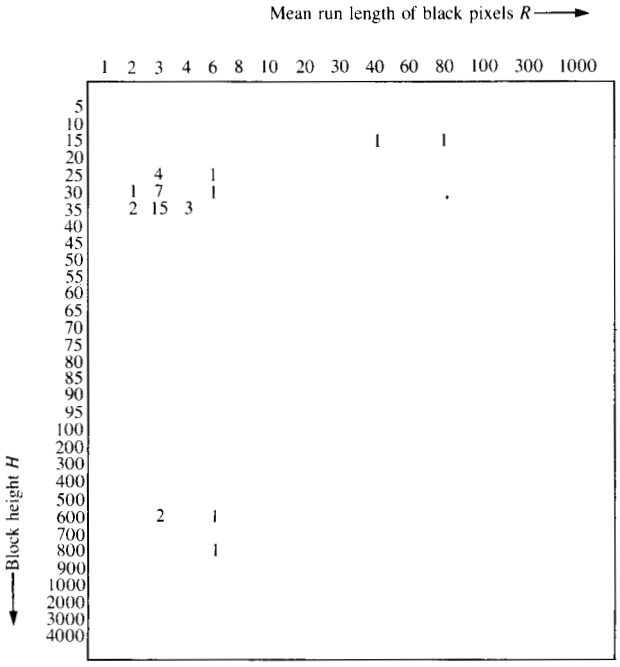

Figure 3 Feature histogram (R-H plane) of the document shown in Fig. 2.

assignments are made quickly, while the difficult ones are deferred to a more complex process, is a promising approach to analyzing the immense amount of data in a scanned document.

Recognition of text data

Hundreds of font styles and sizes are available for the printing of documents. Mathematical symbols, foreign characters, and other special typographic elements may also appear. The tremendous range of possible input material for the Document Analysis System makes it impractical to attempt to design completely automatic recognition logic. Instead, an algorithm is used to collect examples (called "prototypes") of the various patterns in the text.

The procedure, known as "pattern matching," is conducted as follows. The text is examined, character by character, in left-to-right, top-to-bottom sequence. Two output storage areas are maintained, one to hold the prototypes and the other to record for each text pattern both its position and the index of a matching prototype. The positions saved can be, for example, the coordinates of the lower left corner of the array representing the pattern.

Initially the prototype set is empty; thus, there can be no matching prototype for the first input pattern, and it is stored as the first prototype. In addition, its position and the index "1" are placed in the second output area. The second character is then compared to the first by means of a correlation test (for which one of many variants may be used). If it matches, then its position and the index 1 are recorded. If there is no match, it is added to the library as a second prototype. The third character is then read and a match sought from among the members of the library. This operation is repeated for each input pattern, in turn, with the pattern being added to the prototype collection whenever a match cannot be found.

Pattern matching reduces the recognition problem to one of identifying the prototypes, since the prototype character codes can be substituted for the sequence of indexes obtained for the text. The codes and accompanying positions constitute a document encoding that is sufficient for creating a computerized version of the original text or for conducting information retrieval searches within its content. Other merits of the approach are that (1) the pattern-matching scheme is simple to implement, (2) it permits the entry of a wide variety of font styles and sizes, (3) the recognition logic need not be designed in advance and no design data need be collected.

The prototypes can be identified in several ways. If the document is known by the user to have been printed in any of a number of specified fonts, then prestored OCR recognition logic may be applied [11]. Text in a given language source can be partially recognized using dictionary look-up, tables of bigram or trigram frequencies, etc.; OCR errors can be corrected using such context aids as well. Finally, even if automatic techniques cannot be used, the prototypes can be displayed at a keyboard terminal and identified interactively by the user. The number of keystrokes needed to identify the prototypes is an order of magnitude less than that required to enter typical documents manually.

Pattern matching has previously been used by Nagy [1] for text encoding and by Pratt [12] for facsimile compression. The technique used for pattern matching in both of these systems, however, suffers from one major defect, particularly when the algorithm is implemented by software that is to run on a serial processor. The problem is that each text character is compared pixel-by-pixel with all of the prototypes that are close to it in simple measurements such as width, height, area, etc. As the size of the library increases, many prototypes may qualify, and the time required to do the correlations with a conventional sequential instruction computer can be excessive.

To reduce the amount of computation, the Document Analysis System employs a preclassifying technique described in more detail in [13]. The preclassifier is a decision network whose input is a pattern array and whose output is the index of a possible matching prototype pattern.

Each interior node is a pixel location, and each terminal node is a prototype index. Two branches lead from each interior node. One branch is labeled "black," the other, "white." Starting with the root a node is accessed, and the pixel location that it specifies is examined in the input array. The color of this pixel determines the branch to the next node. When a terminal node is accessed, the corresponding prototype index is assigned to the input.

Each successive input array is first classified by this network, which examines only a small subset of the pixels before producing an index. The network is initially null, but it is modified each time a new prototype is added to the library. The modification consists of adding an extension to the decision tree (or network) that permits it to distinguish the new prototype as well as all previously selected prototypes.

The preclassification is either verified or nullified by comparing the input pattern with the prototype selected by the network. The two-step operation is repeated with the input array shifted into several registration positions until either a matching prototype is found, or else none is found and the input is designated as a new prototype.

The advantage of this approach is that only one pattern correlation is needed to classify an input in each registration position used. Thus, the classification is greatly accelerated compared with straightforward pattern matching. The reduction in classification time must be balanced against the time expended in modifying the decision network each time a new class is found and against the possibility of creating extra classes due to preclassification errors made by the network. Considerable effort has gone into developing an efficient network adaptation algorithm.

The design of the preclassifier is based on a scheme in which the pixels of each prototype are categorized as "reliable" or "unreliable." A pixel is defined to be "reliable" for a given prototype if its four horizontal and vertical neighbor pixels are of the same color. Using this categorization, the decision network is constructed such that for any input pattern the sequence of pixels examined have the following properties. First, every pixel examined has the same color in both the input array and in the selected prototype. Second, if these pixels in the input are compared with the corresponding positions in any of the other prototypes, at least one pixel will be both different in color in the two patterns and a "reliable" pixel for the prototype. In this way, the preclassification is based on pixel comparisons away from the edges, where most variation occurs in printed characters.

Three methods of designing the preclassifier have been investigated. The top extension approach (called strategy A)

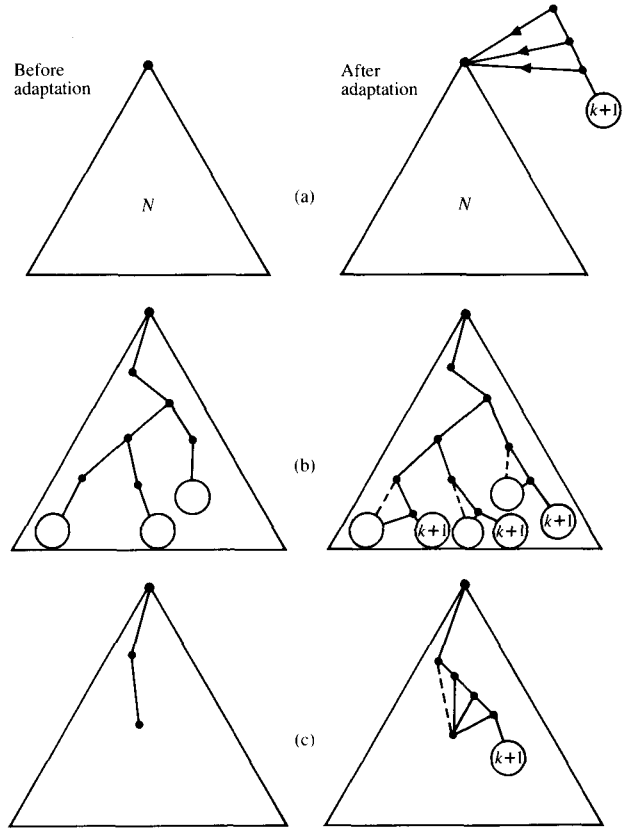

Figure 4 Strategies for modifying the decision network to include class $k + 1$; (a) top extension; (b) bottom extension; and (c) critical mode extension.

adds pixels to be examined above the root node of the previously designed decision network. The strategy is to examine one or more "reliable" pixels such that the added network can distinguish the new prototype class $G(k + 1)$ from the remaining k classes. [See Fig. 4(a)]. The bottom extension approach (called strategy B) retraces the path of $G(k + 1)$ through the decision network as in preclassification, but with the difference that when a pixel that is unreliable in $G(k + 1)$ is encountered, both branches from the node are followed. Thus, the extension routine traverses multiple paths through the network. Each terminal node reached in this way is then replaced by a three-node subgraph. The top node of the subgraph specifies a pixel that is a "reliable" pixel for $G(k + 1)$, and is "reliable" and of opposite value for the prototype P, identified by the terminal node. The branches from this node lead to terminal nodes identified with P and with $G(k + 1)$, respectively. The extension procedure is repeated for each terminal node reached, as illustrated in Fig. 4(b).

The third approach (called strategy C) is called "critical node extension." Here, prototype $G(k + 1)$ is entered into

Figure 5 Prototype patterns from the document shown in Fig. 2(a).

network $N(k)$ as in the bottom extension strategy, and its path is traced until it reaches a node n specifying an unreliable pixel in $G(k + 1)$. At this point it is determined which subset of prototypes $G(1)$, $G(2)$, $\cdots$, $G(k)$ can also reach node n. Call this prototype subset H. The branch into node n is reconnected to a subnetwork consisting of a sequence of pixels that are reliable in $G(k + 1)$ and that discriminate $G(k + 1)$ from members of H. The specification of this subnetwork is identical to strategy A except that only members of H, rather than all the prototypes, are discriminated from $G(k + 1)$. The critical node extension is illustrated in Fig. 4(c).

Each strategy has its advantages and limitations. Strategy A is simplest in concept. Each time a new class is added, a local modification to the network is made at its root. The disadvantage of the approach is that, as more and more classes are added, the structure resembles a chain of subnetworks, each link of which corresponds to a class. If a pattern belonging to a class near the bottom of the chain is entered into the network, then one or more pixels must be examined at each of the higher links. Since the links require more nodes as the number of classes increases, the time required to classify a pattern grows more than linearly with the number of classes.

Strategy B, on the other hand, results in a relatively balanced structure. Starting from a null network, strategy B produces a tree graph, since only subtrees are appended at each step. The exact shape of the tree depends on the prototype pixels, but in general it has a shorter average path length than the network obtained using strategy A. In fact, since any path is increased in length by at most one node each time a class is added, then a network designed using strategy B can have no path longer than K nodes when it has been extended to accommodate K prototypes, and typically the growth in average path length is approximately logarithmic.

However, strategy B calls for modifying the network in a number of locations. As the network grows, many terminal nodes may have to be extended in order to add a single class. This effect increases both the time required for adaptation and the storage requirement for the network.

Strategy C offers a compromise between the other two approaches. The network is modified only at a single node each time a class is added, but the strategy offers the potential of a more balanced configuration than the chain produced by top extension. In a particular case, if each successive prototype followed a path consisting only of interior pixels, then only terminal nodes would ever be modified, and strategy C would be equivalent to strategy B. On the other hand, if for each successive extension the root node specifies an edge pixel for the new prototype, then strategy C yields the same chain configuration obtained using strategy A. In practice, the results lie between these extremes, as shown below.

Figure 5 shows the prototype patterns using the proposed text data recognition method on the document shown in Fig. 2(a). While there is some redundancy in the prototype set, only a small part of this is due to preclassifier errors; the major portion is due to failures of the correlation stage to declare similarity between an input and a prototype of the same class. (The threshold for declaring a correlation match has to be set rather tightly to avoid giving different character types the same designation, a more serious error.) A comparison with a conventional pattern-matching algorithm yielded about the same size prototype set. The efficiency advantage of the preclassifier system is indicated in the plot of Fig. 6, in which a typewritten document was the input. An average of only 130 pixels per character pattern was examined by the preclassifier (out of approximately 1000 in each pattern

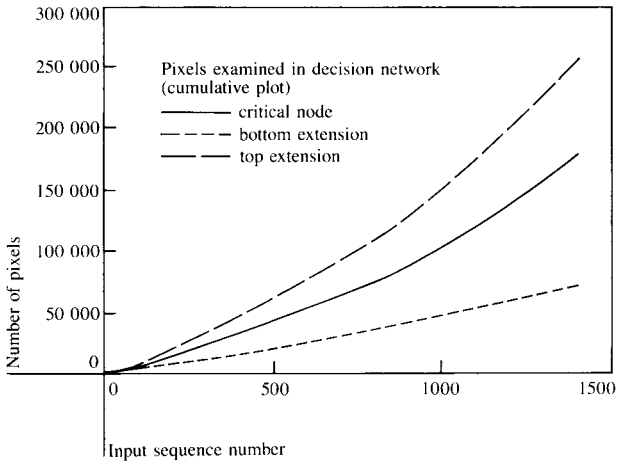

Figure 6 Efficiency advantage of preclassifier.

array) using the critical node extension method. In this same experiment, appreciably more CPU time was spent in using the decision network for classification than in constructing it. In the least favorable case, bottom extension, the ratio of preclassification time to adaptation time was 3:1; for the critical node method the ratio rose to 10:1. These data were obtained from simulation experiments programmed in APL and do not provide estimates of the timings obtainable with efficient programming of the method. An effort is currently under way to reprogram the algorithm and to optimize its speed.

A side benefit of the preclassification approach has been in a recursive technique for the segmentation of touching character patterns (described in detail in [14]). The proper segmentation of closely spaced printed characters is one of the major problems in optical character recognition. The Document Analysis System combines segmentation with the classification of the component patterns. Following the pattern-matching stage, each prototype wider than twice the width of the narrowest prototype undergoes a series of trial segmentations. The two-storage preclassification/correlation technique attempts to find matching prototypes for the trial segments. The recursive algorithm segments a sequence of touching characters only if it can decompose the segment into patterns that positively match previously found prototypes. Because the number of patterns tested for segmentation is small and because the use of the preclassifier makes the process efficient, the computational overhead introduced by recursive segmentation is relatively insignificant. Figure 7 shows a sampling of composite patterns successfully segmented by this technique.

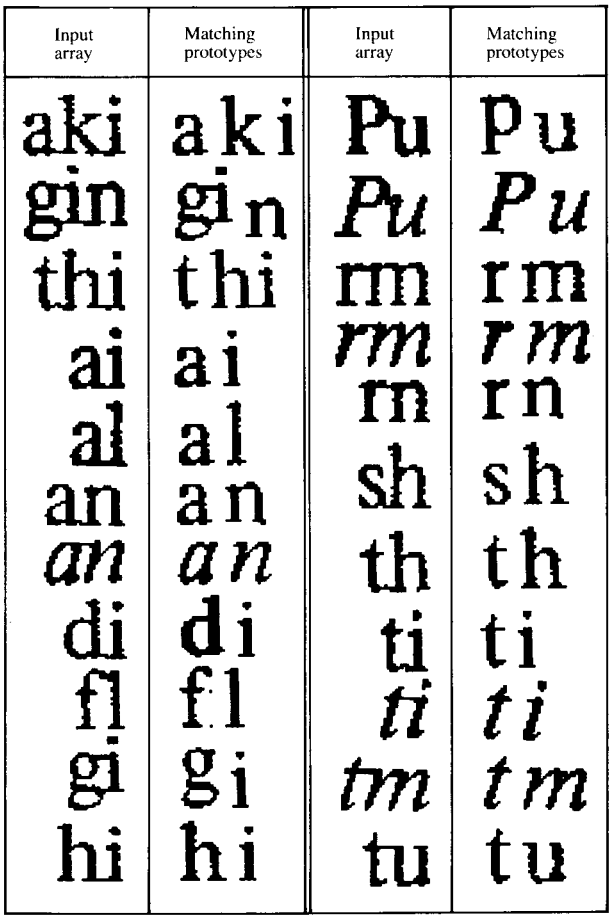

Figure 7 Illustration of the recursive touching character segmentation method.

Conclusion

This paper has described some of the requirements for and a system overview of a document analysis system which encodes printed documents to a form suitable for computer processing. Not all the system components mentioned have been designed and built. The initial work on the project has focused on techniques for the separation of a document into regions of text and images and on the recognition of text data.

The image segmentation method has been tested with a variety of printed documents from different sources with one common set of parameters. Performance is satisfactory although misclassifications occur occasionally. With the user in an interactive feedback loop, misclassification errors can be checked and corrected quite readily in the classification table. A more sophisticated pattern discrimination method [10] is being tested to further classify those blocks that are labeled as non-text.

Pattern matching has been investigated as a fundamental approach to reducing a large number of unknown character patterns to a small number of prototypes. A new, highly

efficient approach to perform the matching has been implemented by employing an adaptive decision network for preclassification. The reduction of computational load is very important when processing is done by a conventional computer. Preliminary results on trial documents have yielded prototype sets of low redundancy, as well as low rates of misclassification.

Acknowledgments

We wish to acknowledge Y. Takao of IBM Tokyo Scientific Center in Japan for his contribution in writing a package of image-editing software (called IEDIT) for our Document Analysis System. We also appreciate the stimulating discussions with G. Nagy during his visit to our laboratory.

References

1. R. N. Ascher, G. M. Koppelman, M. J. Miller, G. Nagy, and G. L. Shelton, Jr., "An Interactive System for Reading Unformatted Printed Text," *IEEE Trans. Computers* **C-20,** 1527–1543 (December 1971).
2. A. Steinbach and K. Y. Wong, "An Understanding of Moiré Patterns in the Reproduction of Halftone Images," *Proceedings of the Pattern Recognition and Image Processing Conference,* Chicago, Aug. 6–8, 1979, pp. 545–552.
3. G. Nagy, "Preliminary Investigation of Techniques for Automated Reading of Unformatted Text," *Commun. ACM* **11,** 480–487 (July 1968).
4. E. Johnstone, "Printed Text Discrimination," *Computer Graph. & Image Process.* **3,** 83–89 (1974).
5. F. M. Wahl, K. Y. Wong, and R. G. Casey, "Block Segmentation and Text Extraction in Mixed Text/Image Documents," *Research Report RJ 3356,* IBM Research Laboratory, San Jose, CA, 1981.
6. F. Wahl, L. Abele, and W. Scherl, "Merkmale fuer die Segmentation von Dokumenten zur Automatischen Textverarbeitung," *Proceedings of the 4th DAGM-Symposium,* Hamburg, Federal Republic of Germany, Springer-Verlag, Berlin, 1981.
7. L. Abele, F. Wahl, and W. Scherl, "Procedures for an Automatic Segmentation of Text Graphic and Halftone Regions in Documents," *Proceedings of the 2nd Scandinavian Conference on Image Analysis,* Helsinki, 1981.
8. F. M. Wahl and K. Y. Wong: "An Efficient Method of Running a Constrained Run Length Algorithm (CRLA) in Vertical and Horizontal Directions on Binary Image Data," *Research Report RJ3438,* IBM Research Laboratory, San Jose, CA, 1982.
9. A. Rosenfeld and A. C. Kak, "Digital Picture Processing," Academic Press, Inc., New York, 1976, pp. 347–348.
10. F. M. Wahl, "A New Distance Mapping and its Use for Shape Measurement on Binary Patterns," *Research Report RJ 3361,* IBM Research Laboratory, San Jose, CA, 1982.
11. R. G. Casey and G. Nagy, "Decision Tree Design Using a Probabilistic Model," *Research Report RJ 3358,* IBM Research Laboratory, San Jose, CA, 1981.
12. W. H. Chen, J. L. Douglas, W. K. Pratt, and R. H. Wallis, "Dual-mode Hybrid Compressor for Facsimile Images," *SPIE J.* **207,** 226–232 (1979).
13. R. G. Casey and K. Y. Wong, "Unsupervised Construction of Decision Networks for Pattern Classification," *IBM Research Report,* to appear.
14. R. G. Casey and G. Nagy, "Recursive Segmentation and Classification of Composite Character Patterns," presented at the 6th International Conference on Pattern Recognition, Munich, October 1982.

Received May 3, 1982; revised July 7, 1982

Richard G. Casey *IBM Research Division, 5600 Cottle Road, San Jose, California 95193.* Dr. Casey has worked primarily in pattern recognition, particularly optical character recognition, since joining IBM in 1963. Until 1970, he carried out projects in this area at the Thomas J. Watson Research Center in Yorktown Heights, New York. After transferring to San Jose, he worked initially on the analysis of data bases, but returned to the character recognition area in 1975, soon after completing a one-year assignment for IBM in the United Kingdom. Dr. Casey received a B.E.E. from Manhattan College in 1954 and an M.S. and Eng.Sc.D. from Columbia University, New York, in 1958 and 1965. From 1957 to 1958, he was a teaching assistant at Columbia University and from 1958 to 1963, he investigated radar data processing techniques at the Columbia University Electronics Research Laboratories. While on leave of absence from IBM in 1969, he taught at the University of Florida.

Friedrich M. Wahl *Lst. s Nachrichtentechnich Technische Universitaet, Arcisstrasse 21, 8 Munich 2, Federal Republic of Germany.* Dr. Wahl studied electrical engineering at the Technical University of Munich, where he received his Diplom and his Ph.D. in 1974 and 1980. In 1974 he started a digital image processing group at the Institute of Communication Engineering at the Technical University of Munich. Dr. Wahl worked in the image processing project in the Computer Science Department at the IBM Research laboratory in San Jose, California, for one year, 1981 to 1982, as a visiting scientist. At San Jose, he worked on problems of document analysis systems, image segmentation, shape description, clustering algorithms, and line drawing decomposition. He has also been involved with automatic image inspection for manufacturing. Dr. Wahl is a member of the Institute of Electrical and Electronics Engineers.

Kwan Y. Wong *IBM Research Division, 5600 Cottle Road, San Jose, California 95193.* Dr. Wong received his B.E. (with honors) and his M.E. in electrical engineering from the University of New South Wales, Sydney, Australia, in 1960 and 1963. Since he received his Ph.D. from the University of California, Berkeley, in 1966, he has been working at the IBM Research laboratory in San Jose. He was the leader on projects in process control systems, air-jet guiding for flexible media, and image processing. Currently he is the manager of the image processing project in the Computer Science Department. His recent research activities are in the areas of image processing and pattern recognition systems for office automation, manufacturing automated visual inspection systems, special image processing hardware, and algorithms.