Segmentation Methods for Character Recognition: From Segmentation to Document Structure Analysis

HIROMICHI FUJISAWA, MEMBER, IEEE, YASUAKI NAKANO, MEMBER, IEEE, and KIYOMICHI KURINO

Invited Paper

This paper discusses character segmentation methods, a key technology for character recognition that determines the usability and applicability of optical character readers. A pattern-oriented segmentation method that leads to document structure analysis is presented. A first example of advanced character segmentation is touching handwritten numeral segmentation. Connected pattern components are extracted instead of a pixel image, and spatial interrelations between components are measured to group them into meaningful character patterns. Stroke shapes are analyzed in the case of touching characters. A method of finding the touching positions can separate about 95% of connected numerals correctly. Ambiguities are handled by multiple hypotheses and verification by recognition. An extended form of pattern-oriented segmentation is also discussed by presenting another example of tabular form recognition. Document images of tabular forms are analyzed, and frames in the tabular structure can be extracted. By identifying semantic relationships between label frames and data frames, information on the form can be properly recognized. Advanced character segmentation with a document structure analysis capability is becoming increasingly significant in automating information extraction from various kinds of documents.

Keywords—*Character segmentation, touching character, contour analysis, multiple hypothesis, form understanding, document analysis, document filing.*

I. INTRODUCTION

Pattern recognition has three main steps: observation, pattern segmentation, and pattern classification. Segmentation is the step in which observed patterns are segregated into units of patterns that seem to form characters. Pattern classification is the process of classifying such a unit of patterns. Identifying such patterns among many patterns in sight is difficult, however, since proper pattern segmentation requires prior knowledge of which patterns form a meaningful unit. This means that, although pattern recognition requires pattern segmentation as a previous step, pattern segmentation itself requires a pattern recognition capability.

Character recognition has one of the longest histories of commercial products for pattern recognition applications. This has been possible because environmental conditions for optical character readers (OCR's) could be set artificially. In the case of printed characters, standard fonts were designed at an early stage. With regard to handwritten characters, restrictions on the style of writing could be posed. Even a standard guideline for writing style was created in Japan. Although these restrictions have been relaxed with advances in character recognition, these environmental conditions are still important in practical character recognition. This is in contrast to speech recognition, where it is difficult and impractical to constrain people to speak in a certain manner.

Solving the character segmentation problem is one of the keys to putting character recognition technology to practical use. Until recently, this problem has been avoided by special formats on OCR forms and by requests that users write characters carefully in separate boxes. Such conditions have been obstacles to widening the OCR market, however. This is especially true for handwritten character recognition. In order to introduce OCR's for data entry in offices, special forms that are thicker and larger than normal forms and that must be handled carefully have been designed. Users are required to train themselves to handwrite characters in a certain manner. These restrictions have impeded the introduction of OCR's, although they make character recognition practical.

Such constraints have been eased by advances in device technology, computer technology, and character recognition

Manuscript received March 6, 1991; revised November 12, 1991.

H. Fujisawa is with the Central Research Laboratory, Hitachi, Ltd., 1-280 Higashi-Koigakubo, Kokubunji, Tokyo, 185 Japan.

Y. Nakano is with Shinshu University, 500 Wakasato, Nagano, 380 Japan.

K. Kurino is with the Odawara Works, Hitachi, Ltd., 2880 Kohzu, Odawara, Kanagawa, 256 Japan.

IEEE Log Number 9202523.

technology. This paper briefly reviews the technological advances that have affected OCR specifications and functionality. Then, advanced character segmentation methods are presented in detail, because this problem has not always been given a high priority. Several recent advances are covered, among them touching character segmentation, which allows relaxation of constraints in OCR forms and writing styles. The technology presented is already being applied to commercial products.

This paper also looks into an area of document structure analysis which is considered to be an extension of character segmentation. A new methodology learned in advanced character segmentation, namely pattern orientation, is one of the keys in document structure analysis. As an example, a method of tabular form understanding is presented. Forms with a tabular structure printed in nondropout color are analyzed, and frames for labels and data can be identified automatically.

This area is increasing in importance in applying character recognition technology to office automation applications. One such new application is optical document filing. Tens of thousands of document images can be stored on optical disks and retrieved when required. Document structure analysis and character recognition can be applied to automate information extraction, especially indexing. A system image of automated document filing, a futuristic system image, is also discussed.

II. Historical Background of Character Segmentation for OCR's

A. Segmentation in Fixed Positions

Image sensors have a strong influence on the method of segmentation. First-generation OCR's, in the 1970's, employed a flying spot scanner which used a cathode ray tube (CRT) and photomultipliers, especially for very high speed OCR's. A flying light spot from the CRT is focused on the document. The reflected light is gathered and sensed by photomultipliers. Because of the limited memory capacity in those days, only image data for a single character area were entered into the recognition processor at one time. Since the scanning positions could be controlled dynamically, positions of characters to be recognized could be identified by prescans searching for reference marks and/or patterns.

An image scanner using a laser was also used in first-generation commercial OCR's. The scanner had a polygonal mirror that rotated at a very high speed and reflected the laser beam to scan the paper vertically over the height of one character. Horizontal scanning was done by another mirror, that changed angle once per character line. Reflected light was sensed by a photomultiplier. Using this scanner, the OCR could obtain pixel data from a single line of characters. Positions of character lines were determined by another photosensor detecting reference marks. In those days, OCR forms had such reference marks at one or both edges of the form to show the positions of the lines, as shown in Fig. 1. A mechanical paper feeder fed in the

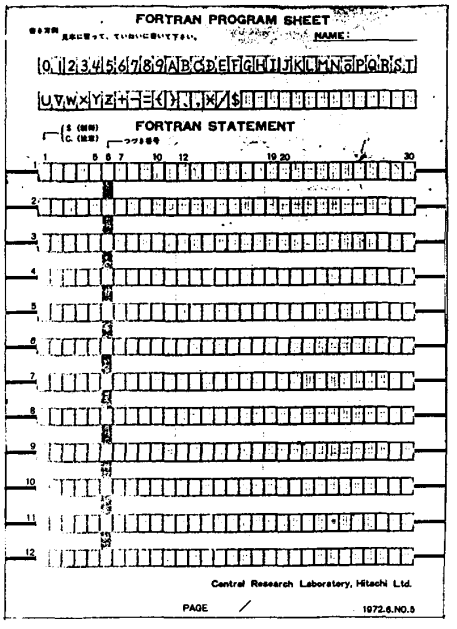


Fig. 1. Reference marks on OCR forms.

form until a reference mark was sensed. The form was moved to the next reference mark after scanning each line. Therefore, the vertical positioning of character lines was done mechanically. Horizontal positions of character fields were calculated by referencing parameters in the OCR memory. The parameters that defined character field positions and their attributes were called format data.

Around 1977, a solid-state image sensor, the charge-coupled device (CCD), appeared and changed the scanner mechanism. The first CCD line sensor had only 128 bits, but replaced the above laser scanning mechanism. The vertical scan was done by the CCD sensor itself, and the horizontal scan was done by mechanical movements of the sensor. The positioning of character lines to be scanned was accomplished by mechanical paper movements, again by sensing reference marks.

In these image scanners, the mechanical paper feeder therefore had to have a high precision to ensure that the calculated character positions coincided with the physical positions. This meant that part of the character segmentation was done by a mechanical paper feeder.

High precision was also required for preprinted forms, in particular for printing alignment and cutting positions. Character boxes were preprinted in a dropout color. Their recommended size was 5 mm by 7 mm, with a gap of 1 mm between boxes, in the case of handwritten character recognition. Positions of character boxes in the scanned image were calculated, and the segregated image data were sent to the recognition processor regardless of the patterns included in the segregated image, as illustrated in Fig. 2. Characters were assumed to be entirely within each physical box and extension of a stroke outside of the box was not allowed. As a result, the total positional accuracy had to be within 0.5 mm.

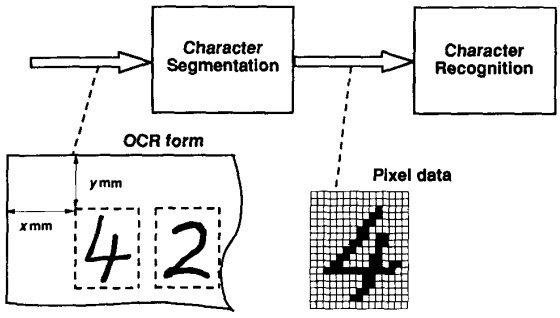


Fig. 2. Position-based pixel-oriented character segmentation.

In other words, the segmentation methods of those days were position based and pixel oriented, in the sense that the segmentation process included only pixel data handling. Extensions to the segmentation method of those days included calculation of a vertical projection profile to make adjustments in horizontal segmentation and checks to include strokes extending out of the box.

B. Full-Page Image Scanning

The emergence of a CCD line sensor with 4096 bits or more, together with a large memory, changed the OCR architecture. The second generation of OCR's started to scan the whole page and in the early 1980's stored the image in a memory. This eliminated the starting and stopping of the paper feeder and increased the total throughput. Forms could run at a very high speed under the long line sensor, which spanned the width of the forms. It became unnecessary to restrict the order of recognition to be from top to bottom, because the entire image was put into an image memory. This allowed more complex form design and more complex applications.

On the other hand, the load for character segmentation increased. Reference marks were no longer necessary. Instead of relying on reference marks, the positions of character boxes had to be estimated from form-edge information in the scanned image. Very fast paper transfer caused skew problems, which required skew measurement and normalization. To make the scanned image outside a form black, the scanner bed under the sensor was normally reflective.

Character segmentation therefore started with form edge detection to identify form position and to estimate skew, by processing the scanned image data. This process determined parameters that were required for estimating character field positions in the memory. Geometrical calculations using these parameters can locate the field positions. Instead of normalizing a skewed image, it was also possible to make an adjustment in locating the character field.

Since the accumulated estimation error was expected to be greater than box gaps, the segmentation method should have relied on pattern information as well. The position-based segmentation was open loop in the sense that estimated position parameters were used with almost no reference to image contents. When estimation got less accurate, fine adjustments in positioning became necessary. To overcome this problem, higher level information such as meaningful patterns had to be incorporated into the segmentation process. This necessitated the introduction of recognition into segmentation.

C. Pattern-Oriented Segmentation Method

A rectangular image area which was a little bigger than the estimated character field was segregated from a full-page image. The segregated image here is called the field image. Instead of extracting a rectangular image area for a single character box, connected pattern components in the field image were extracted first. A contour-following method or a labeling method could be used to extract connected pattern components. A pattern component could be represented in terms of a chain of contour points, or a list of run-length codes. Then, pattern components were sorted from left to right in the field image, and their positions were checked to determine which character boxes they belonged to. Normally, the center position of a pattern component determined the box that a component belonged to. In this way, each character box in a field was assigned to pattern components, which were assumed to form a character pattern.

This approach is different from the pixel-oriented method in the first-generation OCR's. Instead of outputting a rectangular area of pixels, the segmentation process outputs collections of stroke patterns. Pattern components, rather than pixels, are centers of interest. We call this new segmentation method the pattern-oriented method. We developed OCR's that incorporated this method in the early 1980's.

Pattern-oriented segmentation methods opened a new frontier. Overlapping strokes at a box boundary can be properly separated. Conventional pixel-oriented segmentation methods relied on projection profiles of pixel data, and it was difficult to identify overlapping strokes. Further, the new method we have developed can cope with ambiguities. A pattern component whose center is on a boundary between character boxes is determined to belong to neither box. Multiple hypotheses are made instead and verified by the subsequent character recognition process [1]. Connecting and touching handwritten characters can also be handled by analyzing strokes and touching patterns. Character boxes printed in nondropout colors can be processed by analyzing line structures. These new features, some of which have been incorporated into commercial OCR's, are presented in the following section.

The new paradigm of character segmentation that incorporates recognition is also important in cursive script recognition and on-line Chinese character recognition [2], [3].

III. Segmentation Methods for Handwritten Characters

A. Relaxation of Constraints in Handwritten Character Recognition

In analyzing obstacles to broadening OCR markets, we found that artificial restrictions on OCR forms and writing

styles were dominant. The questionnaires and user interviews we made in the early 1980's revealed the following problems. Dissatisfaction of users of conventional OCR's can be summarized as follows. Figures show dissatisfaction scores normalized into a total of 100 points.

1)	Recognition accuracy:	26
2)	Recognition speed:	18
3)	Constraints on writing styles:	13
4)	Lack of intelligent functions:	13
5)	Constraints on character box:	11
6)	Limitation of form paper:	6
7)	Limitation of character sets:	5
8)	Others:	8

Of these problems, the first, the third, and the fifth concern pattern recognition and amount to 50 points.

Interviews with those responsible for the introduction of OCR's revealed different problems. First of all, many things must be prepared for introducing OCR's to new applications. The system designer has to design OCR forms equivalent to original forms. These forms are

1) larger and thicker than the original forms,
2) difficult to redesign from the original forms,
3) unnatural for handwriting in separate boxes,
4) expensive to print with high precision and in multiple colors.

Also, the system designer must enter parameters to define form structures for the OCR system, which is a tedious task.

As a matter of fact, at an office where 46% of the data were entered using OCR's, 21% of the data were still entered by a data entry system (DES) using keyboards, because only 26 out of 300 forms were cost-effective for OCR. Other forms had fewer transactions; it was said that forms with more than 800 transactions (forms) per month were cost-effective for OCR's. This could be observed even from the fact that 70% of the forms used for DES were provisional and made by using word processors for example. Most of them were tabular forms without separate character boxes. Of course, they were printed in nondropout color, mostly in black.

Education and training of the users who were going to fill the forms were also required or recommended. One of the most important OCR applications is entering numerical data into electronic data processing (EDP) systems. A study of character fields in data entry forms showed that 40.9% of the fields were numerical only, 35.8% were alphanumeric, and 3.6% were alphabetical only. About 77% of the fields included numerals. Therefore, we concluded that very high reliability numeral recognition was most important in these applications, especially with relaxed writing restrictions.

The enhancement of second-generation OCR's therefore targeted relaxation in OCR form specifications and numeral writing styles. It should be noted here that writing style relaxation is equivalent to more accurate recognition, because errors and rejections occur in the case of unknown

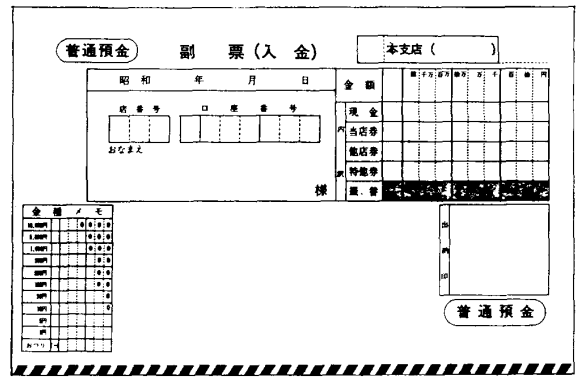


Fig. 3. Example of a non-OCR form: a deposit form used in a Japanese bank.

variations. In other words, the higher the recognition rate required, the more variations that must be recognized. Because not all restrictions could be relaxed, the major ones were selected and targeted as follows.

The form problems described above are closely related. When we analyzed character boxes in the conventional forms, about 30% of the boxes had widths less than 4 mm, and 60% had widths between 4 and 6 mm. This is in contrast to OCR handwritten boxes, which have a width of 5 mm in most cases. As for the height, 47% of character fields were less than 7 mm. Except for OCR forms, there were no cases where character boxes were separate. In other words, there were no gaps between boxes. In some cases, fields took the form of just a single box. Because of the larger box sizes, OCR form structures had to be rearranged. This complicated the form design and caused unnaturalness. Figure 3 shows an example of a non-OCR form which has tabular character fields.

Therefore, the target was to develop a pattern recognition technology that would allow smaller OCR forms. More concretely, it was to reduce the minimum box width from 4.2 mm to 3.5 mm, to reduce the gap from 0.4 mm to zero, and to reduce the line gap from 2.5 mm to zero, with the goal of resolving several interrelated problems. For instance, it was required that there be no major changes in the form design.

When the appearance of revised OCR forms became similar to ordinary forms, it was expected that users might write numerals in a normal way. It was actually not possible to require them to handwrite in the recommended way in boxes such as small as 3.5 mm by 5 mm. It was also expected that adjacent characters might touch each other owing to the small boxes.

Therefore, the real target was to develop an enhanced character segmentation method that could cope with touching handwritten numerals. This required more accurate numerical recognition for less constrained writing styles. Commercial OCR's that allowed the use of small character boxes and tabular fields with high recognition performance, the Hitachi OCR T-550/80 and T-550/27-47, were first marketed in 1983.

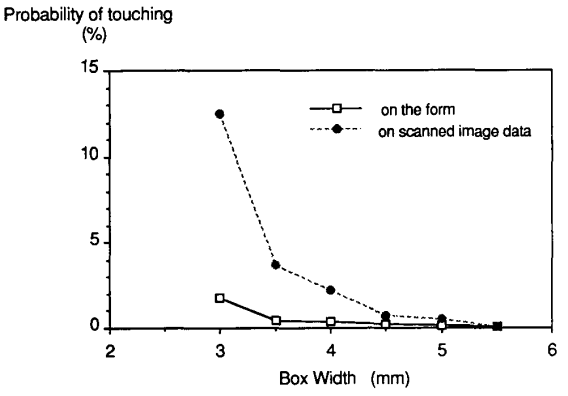


Fig. 4. Probability of touching numerals.

B. Smaller Character Boxes to Reduce the OCR Form Size

It was expected that a smaller OCR form would lead to touching characters. Samples of numerals handwritten in small character boxes were collected, and statistics on touching numerals were analyzed. It was found that there was a steep increase in the frequency of touching when the box was narrower than 3.5 mm, and that more touching patterns were found in the scanned images than on the forms, as shown in Fig. 4. With a 3.5 mm box, about 3.6% of the numerals were touching. Combinations of touching pairs were also studied. The proportions of touching pairs are shown in Fig. 5. When X represents any numeral, 5–X pairs formed the largest proportion, 42% of touching pairs. The second largest group comprised X–0 pairs, with 24% of occurrences.

Typical touching patterns were also investigated and classified. There were three major types, as shown in Fig. 6. The majority took the form of a horizontal stroke extruding to the right to touch the next character. This happened in about 75% of touching cases. By looking into the touched stroke shapes, it was expected that it would be possible to determine the touching position by contour analysis, measuring changes in the stroke width.

One approach for determining the touching position is a vertical projection profile, which is often used in character segmentation [4]. Numbers of black pixels are counted at each horizontal pixel location, and a marginal distribution is calculated. Another method is to combine segmentation with classification by means of an adaptive decision tree [5]. The segmented character image is divided into the first part to be matched with a reference pattern and the residue. If the matching is successful, the process is repeated recursively. However, these approaches for printed characters are not adequate for touching handwritten characters when their strokes overlap with other strokes. This is because the profile does not have information about internal pattern structures. Strokes overlap very often in the case of handwritten numerals, especially when they are tilted. Here, a stroke shape analysis method for identifying touching portions has been developed.

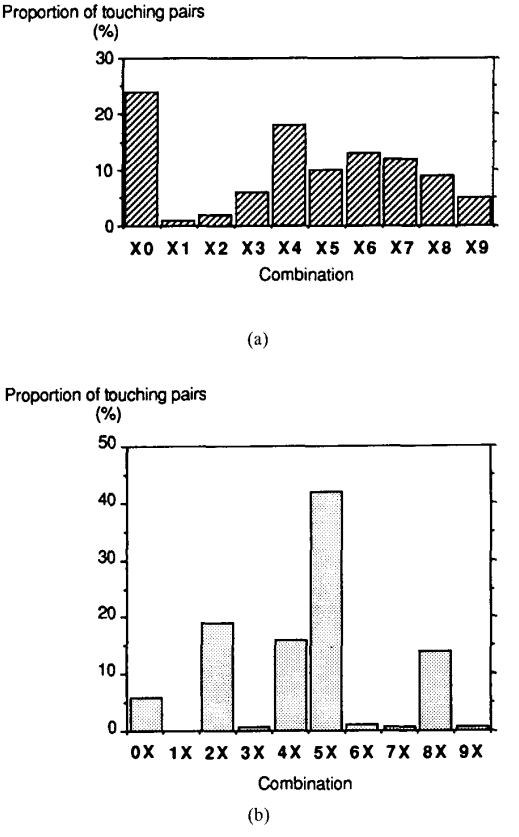


Fig. 5. Frequencies of touching pairs.

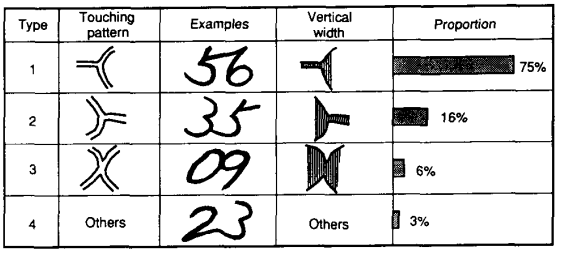


Fig. 6. Typical touching patterns.

C. Pattern-Oriented Segmentation Algorithm for Touching Characters

The principle of the new segmentation method is pattern orientation, as described below. First, by using estimated parameters which define a field image location in the full-page image, contour analysis is applied to the corresponding area of the full-page image to extract connected components. The analysis outputs a set of contours, C, as illustrated in Fig. 7. The contour C_i is a list of contour points, as shown in (2). Here, marker α_i specifies the class of the ith contour; class 0 is an outer loop and class 1 is an inner loop. Another marker, β_i, represents an abnormality. Value 0 is for a normal situation. Values 1 through 4 represent situations in which the corresponding

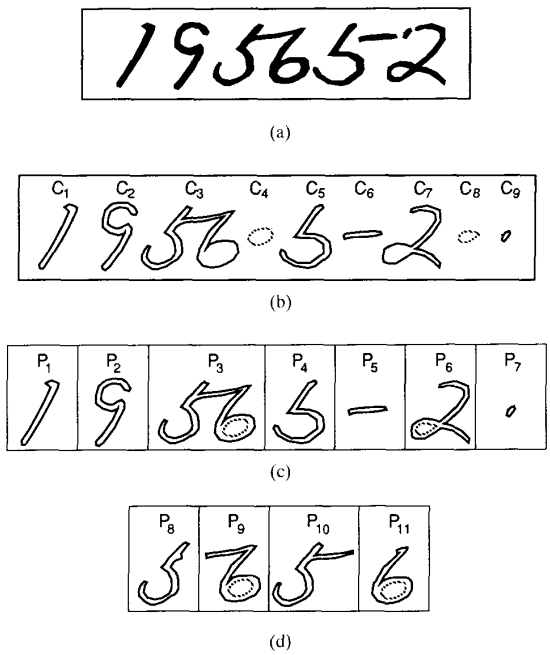


Fig. 7. Pattern-oriented character segmentation. (a) Pixel data. (b) List of contours. (c) List of pattern components. (d) Forcibly separated pattern components.

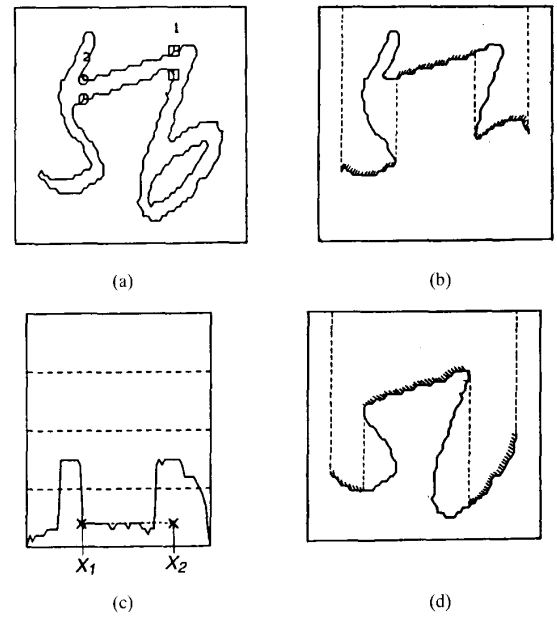


Fig. 8. Measurement of approximate vertical stroke width. (a) Contour of touching characters and candidates for touching points. (b) Upper contour C_U and function $H_U(x)$. (c) Measurement of vertical width $H(x)$. (d) Lower contour C_L and function $H_L(x)$.

connected component extends outside the estimated field area. Each pair (x_{ip}, y_{ip}) represents the x–y coordinates of a contour point, and it is assumed that the contour is followed counterclockwise:

$$C = \{C_i \mid i = 1, 2, \cdots, M\} \tag{1}$$

$$C_i = \{\alpha_i, \beta_i, (x_{ip}, y_{ip}) \mid p = 1, 2, \cdots, N_i\}. \tag{2}$$

where M is the number of contours and N_i is the ith contour's length in number of pixels.

Because the contour information is equivalent to the corresponding binary image, the following segmentation algorithm works on this contour information.

To determine which character box each of M contours belong to, the set C is sorted such that their elements align left to right (Fig. 7(b)). That is, the suffixes i are permuted. Sorting is necessary since contours are not found necessarily from left to right owing to the method of scanning the contour points. For simplicity, we assume that (1) and (2) are already sorted.

A pattern component list, P, is then created, as in Fig. 7(c), where element P_j of P consists of a single outer loop contour and corresponding inner loops if they exist. Namely, list P_j represents a connected pattern component:

$$P = \{P_j \mid j = 1, 2, \cdots, M'\} \tag{3}$$

$$P_j = \{C_k \mid k = K_{sj}, \cdots, K_{ej}\}. \tag{4}$$

Here, the list of contour points C_k is one of the elements of C in (1). Since the list (1) is sorted, C_k for $k = K_{sj}$ is an outer loop, and the following C_k are inner loops. The segmentation process then deals with this component list, without any loss of information.

Before assigning each pattern component to a character box, the size of each component is checked to see if it is a touching pattern. If so, the pattern component needs to be cut into two parts forcibly. The horizontal length (width) of $C_{k_{sj}}$ is compared with thresholds t_{w1} and t_{w2}, where $t_{w1} > t_{w2}$. These threshold values are set from the normal width of a character. The horizontal length is $[\max_p x_{kp} - \min_p x_{kp}]$. If it is greater than threshold t_{w1}, it is determined that the pattern is a touching one. If it is greater than threshold t_{w2}, it is determined that there is a possibility of a touching pattern. Concretely, two hypotheses of nontouch and touch are made in this case. The method of using multiple hypotheses is described later.

In any case, if the horizontal length of the connected pattern component is greater than threshold t_{w2}, the pattern shape is analyzed to detect a touching portion. Touching portions are considered to have a big change in the vertical width of the strokes, as illustrated in Fig. 6.

We have developed an algorithm for detecting candidates of a touching position based on an approximate measure of vertical stroke width. The principle of the algorithm is illustrated in Fig. 8. The following description of the algorithm is for a connected component P_j in (4).

[Algorithm Part I]

Step 1) Separate contour points into the upper and lower part as shown in parts (b) and (d) of Fig. 8. Separation is

done at the leftmost and rightmost points, giving two lists of contour points:

$$C_U = \{(x_{up}, y_{up}) \mid p = 1, \cdots, U\} \tag{5}$$

$$C_L = \{(x_{lq}, y_{lq}) \mid q = 1, \cdots, L\}. \tag{5'}$$

Step 2) Apply a vertical marginal operation for the upper and lower contours. At some x, more than one contour point exists. For such points, only one contour point is selected. For the upper contour, the lowest point is selected, and for the lower contour, the highest point is selected. In parts (b) and (d) of Fig. 8, selected portions of the contour are shaded. As a result, the two lists of upper/lower contour points are converted to single-valued functions, $H_U(x)$ and $H_L(x)$, each of which represents the y coordinate of the contour point at x.

Step 3) Calculate an approximated measure of vertical width $H(x)$:

$$H(x) = \text{abs}\,[H_L(x) - H_U(x)]. \tag{6}$$

Note here that the direction of the y coordinate is downward. In the intervals where $[H_L(x) - H_U(x)]$ is negative, $H(x)$ does not represent the real width of a stroke because of the approximation in step 2. It is an undefined interval in a sense.

Step 4) Search for candidate locations x_n^*, $n = 1, \cdots, N$, where the stroke should be cut. This is done by comparing the vertical width $H(x)$ with a threshold h_t. As shown in Fig. 8(c), this comparison is made in interval $[X_1, X_2]$ to limit the search range. The interval is defined to be the one where touching occurs. Candidate locations are where the curve $H(x)$ and line h_t cross each other. Let the candidate cutting contour points be

$$\{(x_{un}^*, y_{un}), (x_{ln}^*, y_{ln}) \mid n = 1, \cdots, N\}, \tag{7}$$

where subscripts u and l denote upper and lower, respectively. Examples of detected candidates of touching positions are depicted in Fig. 8(a).

Step 5) If N is not zero, cutting contour points in (7) are shifted such that they are closer to the point where the width of the stroke expands abruptly. Otherwise, go to step 6. The width of the stroke at x_n^* can be calculated as $y_{ln} - y_{un}$ and that in the neighborhood can be determined by following the contour point chains in (5). Let the resulting modified candidate points be

$$\{(x_{un}^*, y_{un}^*), (x_{ln}^*, y_{ln}^*) \mid n = 1, \cdots, N\}. \tag{8}$$

Step 6) If N is zero, and if there is more than one inner loop, go to **Algorithm Part II**.

Step 7) If N is zero, and if $H(x)$ is less than a threshold h_t in the whole interval $[X_1, X_2]$, the middle of the interval is set as a candidate touching position and N is set as 1. This case corresponds to type 4 in Fig. 6.

End of Algorithm Part I

The following algorithm is a special treatment of the third type of touching shown in Fig. 6, which involves touching loops as in "0–0," "8–8," and "0–9."

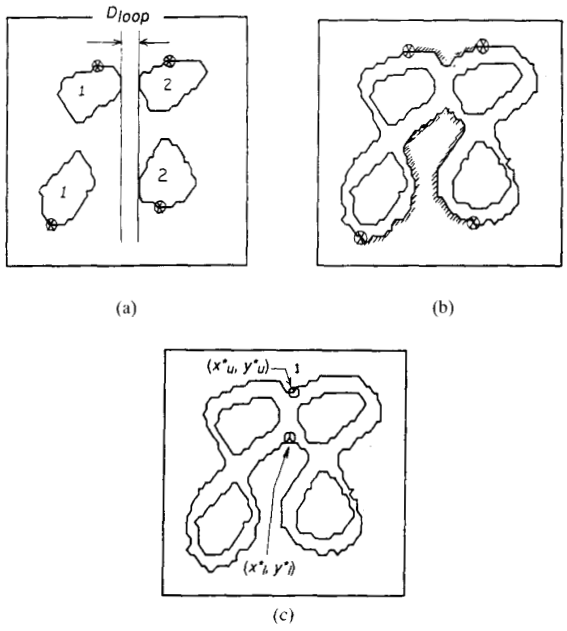


Fig. 9. Forced separation of touching loops. (a) Grouping inner loops. (b) Finding maximal and minimal contour points. (c) Finding the narrowest point.

[Algorithm Part II]

Step 1) Divide the inner loops in (4) into two groups, a left group and a right group as shown in Fig. 9(a), and let D_{loop} be the x distance between the two groups.

Step 2) If D_{loop} is less than a threshold D_t, then set an error flag and exit the algorithm. This is a case where two groups of loops overlap substantially which might not be a touching pair, such as a single "8." Otherwise, go to step 3.

Step 3) For each group of loops, find the highest and lowest inner contour points (Fig. 9(a)). Then, find the corresponding outer contour points, i.e. the two maximal points from (5) and the two minimal points from (5') (Fig. 9(b)).

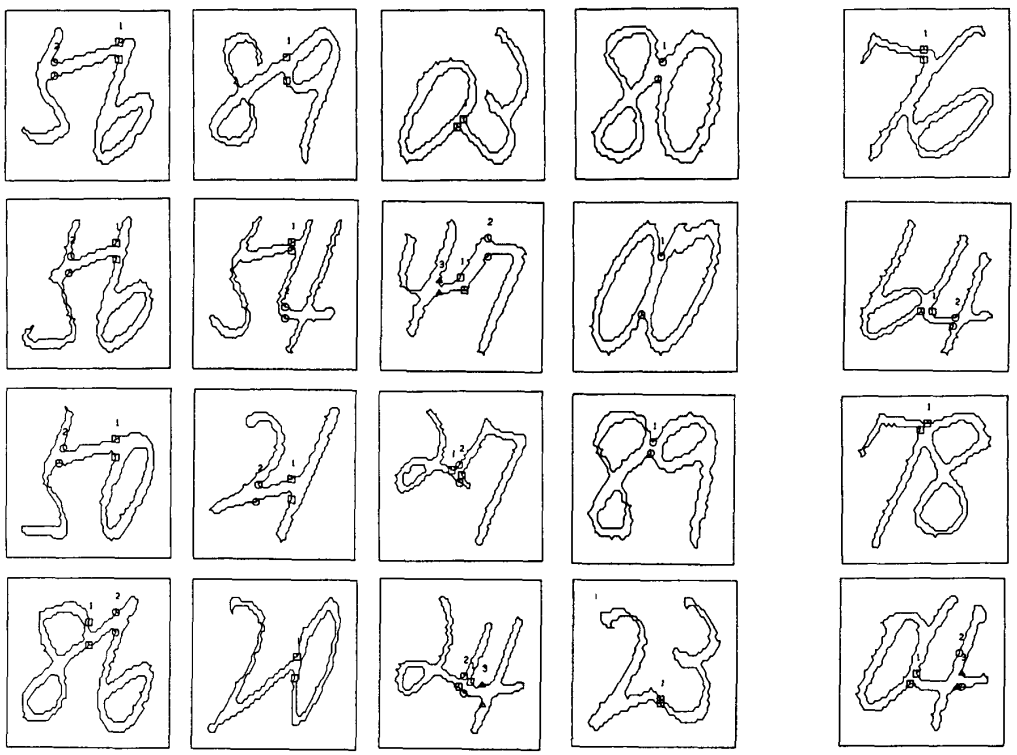
Step 4) Find the lowest contour point (x_u^*, y_u^*) in (5) between the two maximal points found in step 3.

Step 5) Find the highest contour point (x_l^*, y_l^*) in (5') between the two minimal points found in step 3. Then, the cutting points are $\{(x_u^*, y_u^*), (x_l^*, y_l^*)\}$ and N is set to 1 (Fig. 9(c)).

End of Algorithm Part II

When candidate touching positions are identified in terms of pairs of contour points, $\{(x_{un}^*, y_{un}^*), (x_{ln}^*, y_{ln}^*) \mid n = 1, \cdots, N\}$, it is straightforward to cut the connected pattern component into two parts. For each candidate, say $\{(x_u^*, y_u^*), (x_l^*, y_l^*)\}$, a line segment connecting these points is generated. It is represented in terms of a list of point addresses (9), just like contour points:

$$C_B = \{(x_{br}, y_{br}) \mid r = 1, \cdots, R\}, \tag{9}$$



(a) (b)

Fig. 10. Detected touching positions for various patterns. (a) Successful cases. (b) Unsuccessful cases.

where

$$(x_{b1}, y_{b1}) = (x_u^*, y_u^*) \tag{10a}$$

and

$$(x_{bR}, y_{bR}) = (x_l^*, y_l^*). \tag{10b}$$

Then, contour segments C_U and C_L are divided into two parts at the points (x_u^*, y_u^*) and (x_l^*, y_l^*), respectively. When they are written as $C_{U\text{left}}$, $C_{U\text{right}}$, $C_{L\text{left}}$, and $C_{L\text{right}}$, two new detached patterns, P_{left} and P_{right}, are generated as follows:

$$P_{\text{left}} = \{C_{U\text{left}}, C_{L\text{left}}, C_B^-\} \tag{11a}$$

$$P_{\text{right}} = \{C_B, C_{L\text{right}}, C_{U\text{right}}\}, \tag{11b}$$

where the list of contour points in C_B^- is the reverse of C_B. These newly generated patterns are registered in the pattern component list, P, in (3) with additional information showing from which pattern component they originated (Fig. 7(d)).

Examples of detected touching positions in the experiments are shown in Fig. 10. Unsuccessful cases are also shown there. One of the limitations is due to unsuccessful definition of vertical stroke width. Another limitation, which does not appear in the examples, is the case where two characters touch each other at more than one position. This algorithm can be applied to alphabets, although it has not been tested thoroughly.

D. Multiple Hypotheses and Verification by Recognition

A pattern-oriented segmentation method has advantages not only in coping with touching characters but also in handling multiple components. Character patterns are sometimes composed of many connected pattern components. Less constrained numerals have separate components. A separate component sometimes creates ambiguity in deciding which character (or box) it belongs to. This problem can be solved by generating multiple hypotheses [1].

A brief description of the pattern component assignment algorithm is as follows. It is based on grouping adjacent components. Since the list P is already sorted from left to right, the algorithm starts with the first component, P_1, determining whether or not the next component, P_2, is in the same group. To determine this, three measurements on these components are made.

From the two corresponding contour lists in (4), start and end x coordinates, x_{s1}, x_{e1}, x_{s2}, and x_{e2}, are extracted. Then, three measures:

$$d_{ss} = x_{s2} - x_{s1}$$
$$d_{ee} = x_{e2} - x_{e1}$$
$$d_{se} = x_{e2} - x_{s1}$$

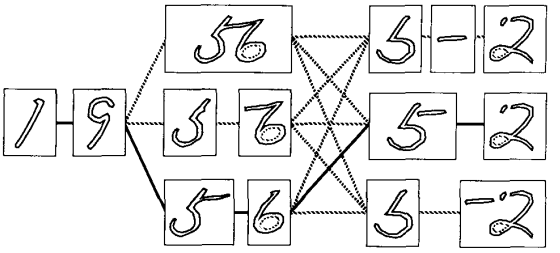


Fig. 11. Network of multiple hypotheses of pattern segmentation.

are calculated. We call these SS analysis (start–start analysis), EE analysis (end–end analysis), and SE analysis (start–end analysis), respectively. For each analysis, these measured values are compared with two preset threshold parameters. Let us describe SS analysis as an example.

Thresholds t_{ss1} and t_{ss2} ($t_{ss1} > t_{ss2}$) are compared with d_{ss}. If $d_{ss} \geq t_{ss1}$, SS analysis outputs an "S" (denoting separate). If $t_{ss1} > d_{ss} \geq t_{ss2}$, it outputs a "U" (undetermined). Otherwise the output is a "G" (group). The final decision is made upon the outputs of the three analyses. There are several ways of combining these local decisions. One is the unanimous decision rule, under which three S's mean separation, three G's mean grouping, and other combinations mean multiple hypotheses of both separation and grouping.

The real segmentation algorithm is more complex because the touching pattern analysis is included. If $(x_{c1} - x_{s1}) > t_{w2}$, the component concerned is assumed to be touching, and the touching pattern analysis, whose basic operations are explained above, is started.

These hypotheses are represented in terms of a network of grouped components as in Fig. 11. The boxes in the figure are hypothesized groups of patterns composed of single or multiple pattern components. Pattern(s) in each group are then fed into the recognition stage, and the character recognition tests the hypotheses. The path with the highest recognition score is selected as the right segmentation. If the selected path has more than one character and the recognition results include a rejection, then all characters on this path are rejected.

E. Evaluation of the Method

To evaluate the developed method, samples of touching numerals were created based on the preliminary analysis of numeral forms. Forty-six combinations which often appeared were extracted, and 20 samples were created for each combination. Some of the touching numeral samples are shown in Fig. 12. Computer simulation showed that about 95% of such touching numerals were correctly segmented and correctly recognized. The rest were all rejected. Figure 13 shows examples of touching numeral recognition simulation. Symbols in the lower-right corner of character boxes are recognition results, including the symbol "??," which means rejection. Small squares in the character boxes highlight the final recognition decision.

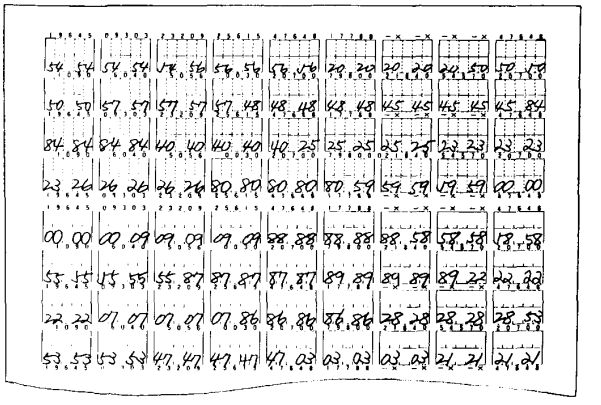


Fig. 12. Samples of touching numerals.

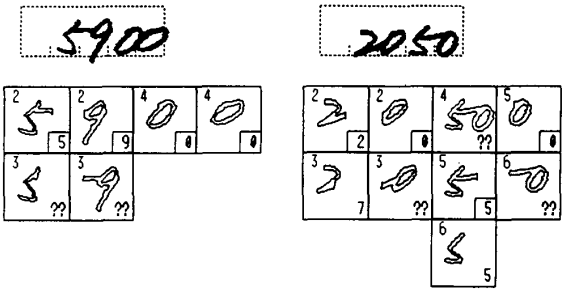


Fig. 13. Touching numeral recognition with multiple hypotheses.

Further evaluation experiments were conducted for an implemented version, which had enhancements in touching position identification. For a larger sample set of about 1 000 000 handwritten numerals, segmentation accuracy was evaluated. Segmentation errors were 0.159%, 0.0394%, and less than 0.01% for 3.0 mm, 3.5 mm, and 4.2 mm character boxes, respectively. For comparison, the segmentation errors with the conventional method were 4.42% for a 3.5 mm box and 0.138% for a 4.2 mm box for the same sample sets. It can be seen that the new method gives 100 times higher reliability than the conventional method for problematic situations.

As described in the previous section, the numeral classification capability was enhanced by adding more template patterns, and patterns that were improperly detached could all be rejected. To make a multiple-hypothesis method work well, high reliability for rejecting uncertain patterns is very important.

IV. Document Structure Analysis

A. Form Structure Understanding

Document structure analysis can be seen as an extension of character segmentation. Position-based, pixel-oriented segmentation methods had limitations in their capabilities, but a new approach, pattern-oriented segmentation, opened

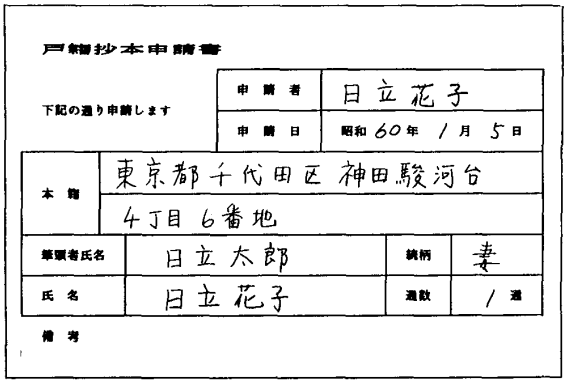


Fig. 14. Application form for Japanese family registry.

new possibilities. Instead of concentrating on extracting character patterns from image data, the structures of printed forms and layouts of character strings in the images need to be analyzed. These new features can open new markets for OCR's.

As discussed before, most data entry sheets were in tabular form and printed in nondropout colors. More interestingly, business forms, for example, are self-explanatory in the sense that the forms have a structure conveying semantic information. They tell the reader what to fill in and where. An example of such forms is shown in Fig. 14, which is an application form for Japanese family registry. It is very effective to identify this semantic structure from scanned form images and to recognize the form contents automatically, because this makes it unnecessary to prepare special OCR forms in advance. We have developed a method for understanding Japanese form structures and have applied it to a prototype system called the document input OCR system (DIOS) [6], [7].

The system has two phases: form registration and form contents recognition. In the form registration phase, it scans unfilled sample forms, and extracts the form structure information to register it into a system file called the form structure information (FSI) file. The FSI includes

1) form identifier
2) input field locations
3) labels for input fields
4) form structure features to identify the form.

This information is equivalent to "format data" for conventional OCR's.

In the form contents recognition phase, the kind of form is first identified by analyzing the form image. It is assumed that there are many kinds of forms. The location parameters of input character fields are then obtained from an FSI file, based on the form identification. Then, images in the character fields are segmented and recognized. Because FSI information includes labels for such fields, proper character sets can be selected for character recognition, and appropriate postprocessing, for example word verification, can be applied. In the case of handwritten Chinese character (kanji) recognition, word matching is very effective for names and mailing addresses. Automatic form registration requires

1) line element extraction
2) frame extraction
3) extraction of spatial relations among frames
4) label recognition
5) extraction of frame interdependency
6) generation of FSI information.

In the first step, horizontal and vertical run-length filtering is applied to the image to enhance horizontal and vertical line elements [8]. It is then reduced in size by an OR mask to connect broken lines and to reduce the amount of computation. Contour analysis is then applied to extract inner loops that represent rectangular areas, which we call frames. After discarding very small loops, four corner points are selected from the contour points of each inner loop. The corner points can be identified by finding points that give max $(x_p + y_p)$, max $(x_p - y_p)$, min $(x_p + y_p)$, or min $(x_p - y_p)$, where $p = 1, \cdots, N$. This is the method used to find the outermost points in handwritten character recognition [9]. By comparing the x and y coordinates of those four corner points, they can be judged as forming a rectangle or not.

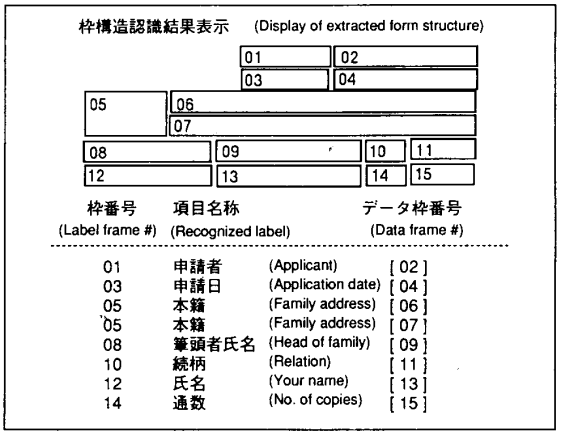
Extracted rectangle frames are ordered and numbered from top left to bottom right. For each frame, a value $x_c + \varepsilon \cdot y_c$ is calculated, where ε is set to about 0.1, and the frames are sorted in increasing order of this value. Here, (x_c, y_c) is a top-left corner point of inner loops. It is possible to give ordered numbers in a reliable way regardless of minor skew and irregularities. A matrix R showing spatial interrelationships between extracted frames is then created. Element R_{ij} is a code representing the spatial relation of frame F_i to frame F_j. R_{ij} can be right-neighbor (RN), rightward (RW), beneath (BN), or below (BW). For example, R_{ij} = RN means that *frame* F_i is the *right neighbor of frame* F_j. Here, "right-neighbor" and "beneath" mean that the two are adjacent. In addition, to be right (below), the frame concerned should have the same or smaller height (width) as the frame to the left (above). These relations can be determined by comparing the addresses of their corner points. Only half of the matrix is filled because the other half is the mirror image. Experimental results of frame extraction and the corresponding spatial relationship matrix are shown in Fig. 15(a), the sample being the one shown in Fig. 14.

Frames are then classified into label and data frames. Label frames are those having patterns in the corresponding frame area, while data frames have no patterns. A special case is a data frame that has characters inside which are units of numbers such as "year," "month," and "day." These data frames can be determined by character recognition to be data frames.

To extract semantic relationships between frames, the characters in the label frames are recognized. The pattern-oriented segmentation method described in the previous section can be used for fields without separate boxes. Characters in the label frames are normally machine-printed

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	*														
2	RN	*													
3	BN	-	*												
4	-	BN	RN	*											
5	-	-	-	-	*										
6	-	-	-	-	RN	*									
7	-	-	-	-	RN	BN	*								
8	-	-	-	-	-	-	-	*							
9	-	-	-	-	-	BW	BN	RN	*						
10	-	BW	-	BW	-	BW	BN	RW	RN	*					
11	-	BW	-	BW	-	BW	BN	RW	RW	RN	*				
12	-	-	-	-	-	-	-	BN	-	-	-	*			
13	-	-	-	-	-	BW	BW	-	BN	-	-	RN	*		
14	-	BW	-	BW	-	BW	BW	-	-	BN	-	RW	RW	*	
15	-	BW	-	BW	-	BW	BW	-	-	-	BN	RW	RW	RN	*

(a)



(b)

Fig. 15. Form structure understanding. (a) Frame interrelationship matrix. (b) Result of form structure extraction.

kanji. They are recognized by a pattern-matching method based on directional feature patterns, which can also be applied to handwritten kanji. The description of the method can be found elsewhere [10]–[12].

The vocabulary of words in labels is limited in such business forms. A list of such labels defines their attributes such as a character set or a word dictionary. For instance, labels such as "applicant," "your name," and "head of family" have an attribute of person's name. In this case, a kanji set for names is selected for recognition and a name dictionary is selected for word matching. Therefore, recognized characters are matched against the vocabulary list, and the corresponding label and attributes are associated with the frame concerned.

Semantic interdependences between label frames and data frames are then determined. For each frame classified as a data frame, a label frame that has RN or BN relation to the frame concerned is looked for by scanning the elements of matrix R. Then, the attributes assigned to the label frame so found are inherited to the data frame, and it is also recorded from which direction it is inherited. If it is not found in the first cycle, a data frame that has an RW or BW relation to the label frame and that has attributes already inherited is looked for. If it is found, the attributes are inherited as before. If there is more than one data frame

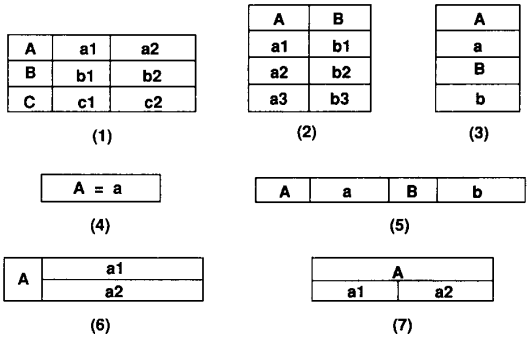


Fig. 16. Recognizable structures of tabular forms.

that meets the conditions, the correct one is selected by taking the inheritance direction into account. Figure 15(b) includes the recognition result of these interdependences. Form structure information is then ready to formulate the frame location parameters, character set name, word dictionary name, etc. This information is used to recognize characters in filled forms.

This algorithm can be applied to a wide variety of tabular forms. The principal structures are shown in Fig. 16, and their combinations are recognizable. Here, capital letters are for label frames and lowercase letters are for data frames. For evaluation purpose, four artificial forms were designed using a word processor and 100 copies were made of each blank form. In form registration experiments, all copies were correctly analyzed in frame extraction and interdependence identification. As for form contents recognition, handwritten kanji were recognized by the DIOS system with an accuracy of about 95%. The word recognition algorithm recently developed by us raised the recognition accuracy to more than 99% by correcting 96.6% of rejections and errors [13].

B. Automated Document Filing and Information Extraction

The importance of document image understanding has increased drastically since an electronic document filing system using optical disks came on the market in Japan [14]. This system can store a huge number of documents as digital images of whole pages. Documents scanned at 400 dots per inch are compressed using the modified modified READ (MMR) coding method, and stored on 5 in. or 12 in. optical disks, which can store up to 20 000 or 200 000 A4-size pages per disk. They can be retrieved for display on a high-resolution monitor and for making a hard copy on a laser printer. This market is growing rapidly, especially in Japan. Although the outlook is promising, one of the problems is in filing of documents. It is a tedious task to place documents on the scanner bed, to start the scanning, to enter keywords or bibliographic information for later retrieval, and to confirm that the quality of the scanned image is adequate.

The concept of automated document filing is a field of research aimed at solving this problem [15]. It includes

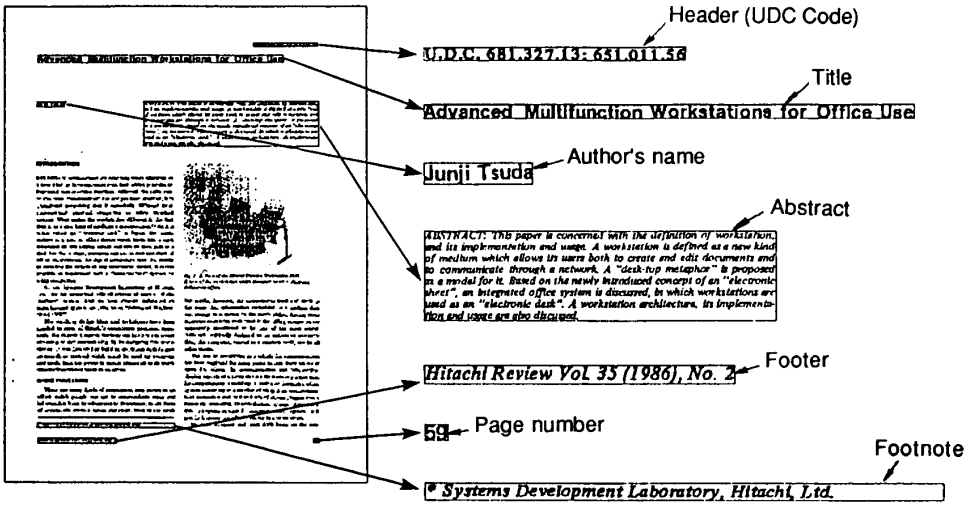


Fig. 17. An example of title page structure analysis.

page class identification (document classification), page number recognition, title page analysis, and extraction of bibliographic information to ease the information input, which is currently done through keyboards. Image processing for automatic page positioning, skew normalization [16], automatic threshold setting for binarization, automatic identification of gray-scale or color picture regions [17], etc., are also important to eliminate the need for human intervention.

An example of title page structure analysis is shown in Fig. 17. We have shown that by incorporating general knowledge about document layout, the physical and logical structure of documents can be extracted [18]. The method we have developed uses a form description language (FDL) to describe the generic layout structure in a framelike representation. Since it is generic in the sense that the descriptions use variables, the prototype system could cope with variations in title length, number of authors, and so on. Spatial interrelations are used to match the knowledge against inputs. An extension of the method has been made to extract the logical structure of chapter–section–paragraph hierarchies [19]. A simplified version of this method can extract bibliographic items from more regular formats [20], [21]. More futuristic systems are planned to convert the scanned document images into hypertext and to allow intelligent browsing and contents retrieval [22].

This research area, document image understanding, has been quite active [23]. The technology can be applied to analyzing documents with very complex structures, such as newspapers [24], [25], mixed picture/diagram documents [26], mail pieces [27], and technical documents with mathematical equations [28]. Common features are the use of document layout knowledge or an expert-system approach [29]–[31], a structure representation in a hierarchical or recursive framework [18], [32], and the use of spatial relationships between pattern elements [18], [33]. It is expected that document image understanding technology will allow a system that can extract important information and knowledge directly from document images.

V. Conclusion

Segmentation methods for character recognition have been discussed. Historically, recognition started with position-based, pixel-oriented segmentation methods, and the current approach is pattern orientation. Pattern-oriented methods have made it possible to segment more flexibly written characters and even touching handwritten characters. Instead of cutting out a section of pixels using projection profiles, connected pattern components are extracted and their shapes are analyzed. Abnormal situations such as touching characters can be handled. For multiple components, spatial relations between patterns are also analyzed to group components. Ambiguities are handled by making multiple hypotheses and verification by recognition. Hypotheses are represented in a network of choices and the best path is selected by character recognition.

Preliminary experiments for touching numeral separation showed that about 95% of touching numeral samples were correctly segmented. For a large size sample set of numerals written in 3.5-mm-wide boxes, the implemented version of the algorithm could segment 99.96% correctly, while a conventional method segmented 95.58% correctly. This new method is applied to commercial OCR's, permitting relaxation of constraints in form design and writing style.

Extended pattern analysis in segmentation allows document structure analysis. Tabular form recognition shows that the preparation of special OCR forms can be eliminated. Another example of title page analysis can automate document filing in the near future. Content structure analysis can be applied to turn plain printed documents into so-called hypertext. This field is becoming increasingly im-

portant and is now being studied by many research groups. In the future, this should lead to automatic knowledge acquisition from documents, or "document understanding."

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Dr. T. Chiba, Senior Chief Research Scientist of Central Research Laboratory, Hitachi, Ltd., for his management of the project. They also wish to thank T. Hananoi, Chief Engineer of Hitachi Electronics Engineering Corporation, and S. Kadota and M. Michino, Odawara Works, Hitachi, Ltd., for their contribution.

REFERENCES

[1] H. Fujisawa and M. Michino, "An augmented segmentation algorithm for connected handwritten numerals," in *Proc. Ann. Conf. Inst. Electron. Comm. Eng. Japan*, 1984, no. 1588.

[2] R. Bozinovic and S. N. Srihari, "Off-line cursive script word recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 68–83, Jan. 1989.

[3] H. Murase, "Online recognition of free-format Japanese handwritings," in *Proc. 9th Int. Conf. Pattern Recognition*, Nov. 1988, pp. 1143–1147.

[4] S. Kahan, T. Pavlidis, and H. Baird, "On the recognition of printed characters of any font and size," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, pp. 274–288, Mar. 1987.

[5] R. G. Casey and G. Nagy, "Recursive segmentation and classification of composite character patterns," in *Proc. 6th Int. Conf. Pattern Recognition*, Oct. 1982, pp. 1023–1026.

[6] Y. Nakano, H. Fujisawa, O. Kunisaki, K. Okada, and T. Hananoi, "Understanding of tabular form documents cooperating with character recognition," *Trans. Inst. Electron. Comm. Eng. Japan*, vol. J69-D, no. 3, pp. 400–409, 1986 (in Japanese).

[7] Y. Nakano, H. Fujisawa, O. Kunisaki, K. Okada, and T. Hananoi, "A document understanding system incorporating character recognition," in *Proc. 8th Int. Conf. Pattern Recognition* (Paris), 1986, pp. 801–803.

[8] K. W. Wong, R. G. Casey, and F. H. Wahl, "Document analysis system," *IBM J. Res. Develop.*, vol. 26, no. 6, pp. 647–656, 1982.

[9] K. Yamamoto and S. Mori, "Recognition of handprinted characters by an outermost point method," in *Proc. 4th Int. Joint Conf. Pattern Recognition* (Kyoto, Japan), Nov. 1978, pp. 794–796.

[10] M. Yasuda and H. Fujisawa, "An improvement of correlation method for handprinted character recognition," *Trans. Inst. Electron. Comm. Eng. Japan*, vol. J62-D, pp. 217–224, 1979 (in Japanese).

[11] O. Kunisaki *et al.*, "Hierarchical structured feature matching method for kanji OCR," in *Proc. Ann. Conf. Inst. Electron. Comm. Eng. Japan* 1985, no. 1535 (in Japanese).

[12] S. Mori, K. Yamamoto, and M. Yasuda, "Research on machine recognition of handprinted characters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 6, pp. 386–405, July 1984.

[13] K. Marukawa, M. Koga, Y. Shima, and H. Fujisawa, "A high speed word matching algorithm for handwritten Chinese character recognition," in *Proc. IAPR Workshop Machine Vision Applications, MVA'90* (Tokyo), Nov. 1990, pp. 445–449.

[14] H. Masuzaki, N. Takahashi, and Y. Kurosu, "HITFILE 650E optical disk filing system," *Hitachi Review*, vol. 38, no. 5, pp. 257–264, 1989.

[15] H. Fujisawa, "Artificial intelligence as applied to optical image filing," in *Proc. Int. Symp. on Optical Memory* (Tokyo), 1987; also in *Japanese J. Appl. Phys.*, vol. 26, supplement 26–4, 1987.

[16] Y. Nakano, Y. Shima, H. Fujisawa, J. Higashino, and M. Fujinawa, "An algorithm for the skew normalization of document images," in *Proc. 10th Int. Conf. Pattern Recognition* (Atlantic City), June 1990, pp. 8–13.

[17] Y. Shima, T. Murakami, J. Higashino, Y. Nakano, and H. Fujisawa, "A segmentation method of color document images for multimedia content retrieval systems," in *Proc. RIAO88 Conference* (Cambridge, MA), Mar. 1988, pp. 1001–1008.

[18] J. Higashino, H. Fujisawa, Y. Nakano, and M. Ejiri, "A knowledge-based segmentation method for document understanding," in *Proc. 8th Int. Conf. Pattern Recognition* (Paris), 1986, pp. 745–748..

[19] H. Yashiro, T. Murakami, Y. Shima, Y. Nakano, and H. Fujisawa, "A new method of document structure extraction using generic layout knowledge," in *Proc. Int. Workshop Industrial Applications of Machine Intelligence and Vision, MIV-89* (Tokyo), Apr. 1989, pp. 282–287.

[20] Y. Nakano and H. Fujisawa, "A method of document understanding for automatic filing," *Trans. Inst. Electron. Comm. Eng. Japan*, vol. J71-D, no. 10, pp. 2050–2058, 1988 (in Japanese).

[21] H. Fujisawa and Y. Nakano, "A top-down approach for the analysis of document images," *Proc. IAPR Workshop Syntactic and Structural Pattern Recognition* (Murray Hill, NJ), June 1990, pp. 113–122.

[22] H. Fujisawa *et al.*, "Document analysis and decomposition method for multimedia contents retrieval," in *Proc. 2nd Int. Symp. Interoperable Information Systems, ISIIS-88* (Tokyo), Nov. 1988, pp. 231–238.

[23] G. Nagy, "At the frontier of OCR," pp. 1093–1100, this issue.

[24] J. Toyoda and Y. Noguchi, "Study of extracting Japanese newspaper article," in *Proc. 6th Int. Conf. Pattern Recognition*, 1982, pp. 1113–1115.

[25] K. Inagaki, T. Kato, T. Hiroshima, and T. Sakai, "MACSYM: A hierarchical image processing system for event-driven pattern understanding of documents," *Pattern Recognition*, vol. 17, no. 1, pp. 85–108, 1984.

[26] L. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, pp. 910–918, Nov. 1990.

[27] C. H. Wang, P. W. Palumbo, and N. Srihari, "Object recognition in visually complex documents: An architecture for locating address blocks on mail pieces," in *Proc. 9th Int. Conf. Pattern Recognition*, 1988, pp. 365–367.

[28] M. Okamoto and A. Miyazawa, "An experimental implementation of document recognition system for papers containing mathematical expressions," in *Proc. IAPR Workshop Syntactic and Structural Pattern Recognition* (Murray Hill, NJ), June 1990, pp. 335–350.

[29] G. Nagy, S. Seth, and S. D. Stoddard, "Document analysis with an expert system," in *Pattern Recognition in Practice II*, E. S. Gelsema and L. N. Kanal, Eds., 1986, pp. 149–159.

[30] A. Dengel and G. Barth, "ANATASIL: A hybrid knowledge-based system for document layout analysis," in *Proc. Int. Joint Conf. Artificial Intelligence*, 1989, pp. 1249–1254.

[31] F. Esposito, D. Malerba, and G. Semeraro, "An experimental page layout recognition system for office document automatic classification: An integrated approach for inductive generalization," *Proc. 10th Int. Conf. Pattern Recognition*, 1990, pp. 557–562.

[32] G. Nagy and S. Seth, "Hierarchical representation of optically scanned documents," in *Proc. IEEE 7th Int. Conf. Pattern Recognition* (Montreal, Canada), 1984, pp. 347–349.

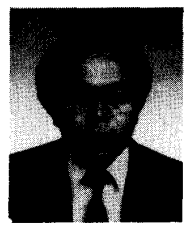
[33] Q. Luo, T. Watanabe, Y. Yoshida, and Y. Inagaki, "Recognition of document structure on the basis of spatial and geometric relationships between document items," in *Proc. IAPR Workshop Machine Vision Applications, MVA'90*, Nov. 1990, pp. 461–464.

Hiromichi Fujisawa (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in electrical engineering from Waseda University in 1969, 1971, and 1975, respectively.

He joined Central Research Laboratory, Hitachi, Ltd., in 1974. There he was engaged in research on speech processing and character recognition for handwritten characters, including kanji's. From 1980 to 1981, he was a visiting scientist in the Computer Science Department of Carnegie Mellon University, Pittsburgh, PA. Since 1983, he has been doing research and development work on an intelligent document filing system, including optical image filing, document understanding, knowledge-based document retrieval, and full-text search machine. Currently, he is a chief engineer at the Software Development Center of Hitachi, Ltd. He has also been a lecturer at the Centre for Informatics at Waseda University since 1985.

Yasuaki Nakano (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in mathematical engineering from the University of Tokyo in 1961, 1963, and 1975, respectively.

From 1963 to 1989, he was with the Central Research Laboratory, Hitachi, Ltd., where he did research on speech synthesis, speech recognition, character recognition, and document understanding. In 1989 he was appointed as a professor in the Department of Information Engineering, Faculty of Engineering, at Shinshu University. Currently, he is responsible for research and teaching in pattern recognition and artificial intelligence.



Kiyomichi Kurino received the B.E. degree from the Mechanical Engineering Department, Kagoshima Technical College, in 1970.

He joined Odawara Works, Hitachi, Ltd., in the same year. Since then, he has been engaged in the development of card punch devices for EDP systems, speech recognition devices, and various versions of OCR's. He has contributed to the development of character segmentation algorithms, kanji recognition algorithms, and document image analysis methods. His technical interests include document understanding, neural networks, and fuzzy logic applications. He is currently an engineer in the Data Entry Device Department of the Odawara Works.