Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 40 (2007) 1855–1867
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

Extraction and segmentation of tables from Chinese ink documents based on a matrix model

Xi-wen Zhang[a, b, *], Michael R. Lyu[b], Guo-zhong Dai[a]

[a] *Laboratory of Human–Computer Interaction and Intelligent Information Processing, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China*

[b] *Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong SAR, China*

Received 10 October 2005; received in revised form 12 May 2006; accepted 25 May 2006

Abstract

This paper presents an approach for extracting and segmenting tables from Chinese ink documents based on a matrix model. An ink document is first modeled as a matrix containing ink rows, including writing and drawing ones. Each row consists of collinear ink lines containing ink characters. Together with their associated drawing rows, adjacent writing rows having an identical distribution of writing lines and\or the same associated drawing rows if available are extracted to form a table. Row and column headers, nested sub-headers and cells are identified. Experiments demonstrate that the proposed approach is more effective and robust.

Keywords: Chinese ink document; Digital ink; Handwriting; Table extraction; Table segmentation

1. Introduction

With the introduction of digital ink capturers such as Anoto Pens [1], MS Tablet PCs [2], IBM ThinkPad TransNote [3], Interactive Pen Displays [4], and Electronic White Boards [5], as well as the techniques of pen-based interaction and interface [6,7], more and more digital ink is being captured. With digital ink capturers, users can intuitively express their intentions in freeform strokes as if they write on paper by pen. The capturers record coordinates, timestamps, and pressures of sampling points for each stroke, and store the entire multi-page document in the ink format. Ink documents represent information with freeform input and employ a new file format. A few Software Development Kits (SDKs) [8,9] have been developed to capture, store, manage, analyze, and recognize ink documents.

An ink document, like other types of documents, may contain various objects such as text, tables, graphics, flow charts, and so on. A table is a compact and efficient presentation format commonly used especially for describing statistical and relational information [10]. It enables readers to search, compare and understand data rapidly [11]. Most ink documents produced today contain various types of tables. Thus, ink document analysis requires the ability to extract and segment tables. Segmented ink tables can be further modified in advanced ways at multiple levels. Some examples are: rows, columns, and cells can be moved interactively; a column or row of headers and cells can be automatically aligned; and tables can be converted into other documentation tools such as MS Word and MS Excel [12].

Tables are first extracted from ink documents, and then segmented. Table segmentation aims to decompose an extracted table into its components at multiple levels (row, column, header, sub-header, and cell), and to identify their relationships. Previous approaches [12] are unable to extract tables with incomplete or no bounding lines. Nor can they extract tables with nested headers and cells accurately. Moreover, they are not able to clearly identify nested headers and cells in a table without bounding and separating lines. In summary, the following situations cannot be handled with

* Corresponding author. Laboratory of Human–Computer Interaction, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China. Tel.: +86 10 62661566-8058.

E-mail address: zxw19@hotmail.com (X.-w. Zhang).

0031-3203/$30.00 © 2007 Published by Elsevier Ltd on behalf of Pattern Recognition Society.
doi:10.1016/j.patcog.2006.05.029

existing algorithms:

(1) No complete bounding lines exist in a table: either there are only two bounding lines located at the top and bottom of a table or there are no bounding lines at all.
(2) A table has no complete separating lines: some separating lines are incomplete or there are no separating lines at all.
(3) A header or cell consists of more than one line of text or contains nothing.
(4) A header or cell might contain Chinese characters, English words, numbers, and special symbols.
(5) There are nested sub-headers and cells. A row header corresponds to multiple nested row sub-headers and cells. Likewise, a column header corresponds to multiple nested column sub-headers and cells.
(6) There are different numbers of components in rows or columns.

To handle the above variations, we observe that, in any table, adjacent rows or their principal parts must have an identical distribution of cells and headers, and have the same associated lines (separating and bounding lines) if available. Based on this observation, we detect ink tables via headers and cells and their associated lines. The matrix model is the closest to abstraction of tables. They both have the same components and structure, such as rows, columns, and elements. An ink document is first modeled as a matrix with ink rows, where some writing rows correspond to rows of headers and cells, and some drawing rows correspond to bounding and separating lines. Chinese documents are well suited to ink input because Chinese characters are more complex than English words in structure and key-in combination. So, we aim to address the real-life problem of adaptively extracting tables from Chinese ink documents; the physical and logical structures of the tables are to be extracted based on a matrix model. It first hierarchically organizes strokes into a matrix to represent an ink document and then extracts sub-matrixes (tables) from the matrix (ink document). An extracted table is segmented into rows, columns, headers, and cells. Extracted and segmented ink tables are stored in the matrix model. Physical structures describe where regions containing table elements are located, and logical structures define the types of these regions and how they form a table [13].

The rest of this paper is organized as follows. Section 2 reviews previous work related to table extraction and segmentation, and presents our analyses of them. Section 3 describes the construction of a matrix for modeling an ink document. Section 4 presents the procedure for extraction and segmentation of ink tables based on the model. Section 5 reports experimental results and performance analyses, and Section 6 draws some conclusions.

2. Related work

A few approaches have been proposed in recent years for extracting and segmenting tables in ink documents in response to the emergence of this new format [8,9]. Jain et al. [12] propose a hierarchical approach for English tables. They apply the restriction that regular tables must contain at least five lines (four boundaries and at least one separating line). Thus, their approach cannot extract tables with incomplete or no bounding lines. Moreover, it consumes much time and memory because it detects lines using the Hough transform. A further issue is that the approach cannot properly extract not regular tables with nested headers and cells because of the use of projection. Cell boundaries are determined from a regular table by its horizontal and vertical lines. Thus, this approach cannot identify headers and cells in a regular table without complete separating lines. The cell boundaries in a not regular table are determined by dividing each line at vertical cell boundaries given by the valleys in the projection histogram. Thus, this approach may fail to identify nested sub-headers and cells in not regular tables.

The task of extracting and segmenting writing from ink documents is related to table extracting and segmenting because tables must contain writing. The approaches used to address this task can be classified into two categories as follows, according to whether or not contexts are used.

(1) Approaches based on histogram projection. Ratzlaff [14] first extracts writing lines from an English ink document based on a histogram projection, then locates words or word groups accordingly. This approach makes rather strong assumptions about document regularity, particularly concerning the line slope and the inter-line gap. It identifies writing lines by a bottom-up approach that takes advantage of spatial and temporal information. It clusters discrete strokes into increasingly larger groups that eventually merge to become complete writing lines. The initial clustering depends on the strong evidence of spatiotemporal proximity. Subsequent merging is based on more sophisticated metrics that include inter-line distances and mean character heights [15,16].

(2) Approaches based on context. More recent studies [12,17] dealing with English ink documents aim to relax these strict constraints and to use contextual information [18]. Jain et al. [12] group writing strokes into lines based on horizontal perspectives and adjacent lines are incrementally grouped. However, the assumption is made that the inter-word distances are more than double the inter-stroke distances within a word. Shilman et al. [17] combine strokes with similar sizes and orientations to form words, lines, and blocks (paragraphs), in a bottom-up order. Blanchard et al. [18] extract paragraphs, lines, and words based on an extension of a probabilistic approach. They take the context into account and consider multiple hypotheses based on Probabilistic Features Grammars. Moreover, they deal with the huge combinatorial complexity by introducing a beam search strategy and by interfacing the parsing algorithm with a genetic algorithm.

Hong et al. [19] propose an approach for extracting characters from Chinese ink documents. Their strategy performs basic and fine segmentation based on varying spacing

thresholds and minimum variance criteria. The five most probable ways are derived and all the possible segments are extracted and recognized. A lattice is created from all the segments and is used to find the most likely character sequence using a Viterbi-based algorithm. It would be better to attain a perfect segmentation of writing by incorporating character recognition, but this requires a perfect character recognizer. The best character recognizer currently available cannot satisfy this requirement. Moreover, the approach has to evaluate recognized characters using syntactic and semantic rules.

Because many tables contain separating and bounding lines, approaches to discriminating writing and drawing strokes are analyzed here. They can be classified into the following two categories according to the features used.

(1) Approaches based on features of an individual stroke. Jain et al. [12] construct a two-dimensional feature space using lengths and curvatures of strokes in an ink document. Its linear decision boundary separates strokes into writing and drawing classes. The features extracted from an individual stroke can provide relevant information to classify the stroke [12], and can give a reasonable initial separation of the two classes [20]. However, to achieve improved performance, it is necessary to take the context of a stroke into account [17,20].

(2) Approaches based on features of multiple strokes. Shilman et al. [17] classify an individual stroke as writing or drawing based on local and global attributes of adjacent strokes, words, lines, and blocks. Bishop et al. [20] discriminate a writing stroke from a drawing one by utilizing not just its own characteristics, but also the information provided by gaps between it and its adjacent strokes, as well as the temporal characteristics of a stroke sequence.

The tool provided by Microsoft Corporation can group strokes into words, lines, paragraphs, and drawings, according to a threshold of line height [8]. However, it cannot identify ink tables. The tool from IBM Corporation is also unable to do so [9]. There are also many other methods for extracting and segmenting tables from documents in images [10,13,21–25], text [26,27], and html [28]. The traditional hierarchical model used in analysis of image documents is to extract multiple levels of components from images. Basic elements in images are pixels. Ink documents can be transformed into images, but at the same time, a lot of information is lost. The lost information is critical to extract components of ink documents. Text and graphics in image documents is formal because they are scanned from printed papers. However, these methods are not applicable to ink table extraction and segmentation [12]. Because ink documents are captured from freehand sketching, ink tables are neither accurate nor formal. They may show a number of deviations, such as: (1) table elements are skew, not aligned; (2) several bounding lines are drawn in a single stroke; (3) a bounding or separating line is drawn in more than one stroke; (4) the elements of the table are written or drawn in a non-consecutive order. Real-life ink documents are unwieldy: most of the documents contain mixed cursive and printed writing, and are often written at angles and in various sizes [17]. Drawings can range from tables and flowcharts to graphics.

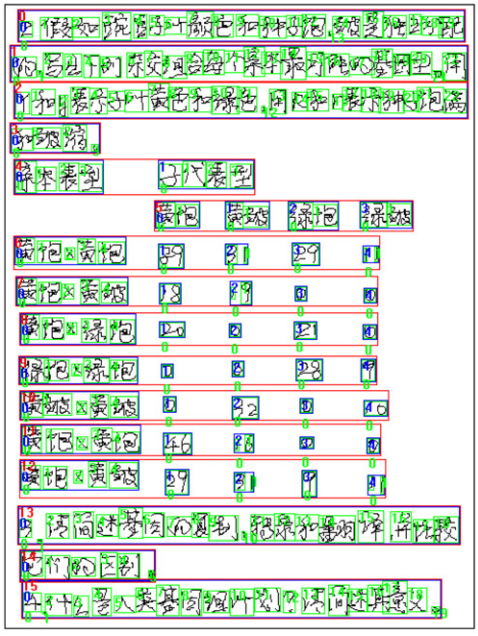
A document can be represented as a hierarchy containing text, tables, graphics, flowcharts and other entities, as well as their components [10,23]. Strokes in an ink document can be grouped as writing characters (words), lines and paragraphs in a bottom-up order [17]. Entities at the same level have four types of relationships: top, bottom, left, and right. All objects in an ink document can be represented as a matrix with multi-level information. The ink matrix can provide additional contextual information to assist in the extraction and segmentation of tables; for example, a table can be identified as adjacent writing rows containing an identical distribution of writing lines and the same drawing rows (if available) associated with them. Based on this observation, we propose to extract and segment ink tables based on a matrix model. It is the abstract model of tables. It consists of rows in horizon and columns in vertical direction. Each row or column includes multiple elements. In summary, our approach is designed to meet the following criteria:

- Ink documents can contain Chinese characters, English words, numbers, and mathematical expressions.
- Tables mixed with Chinese characters, English words, numbers, and special symbols can be processed.
- Tables, with complete, incomplete, or no bounding and separating lines, can be processed.
- Complex logical structures, including sub-nested headers and their cells, and headers and cells with multiple writing lines, can be processed.
- Multi-level contextual information is used to extract and segment tables.
- The a priori knowledge of table formats can be incorporated effectively to improve the reliability and adaptability of table extraction and segmentation.
- The extracted information can be represented as a compact structure for more advanced applications, such as modifying and converting tables into formal ones.

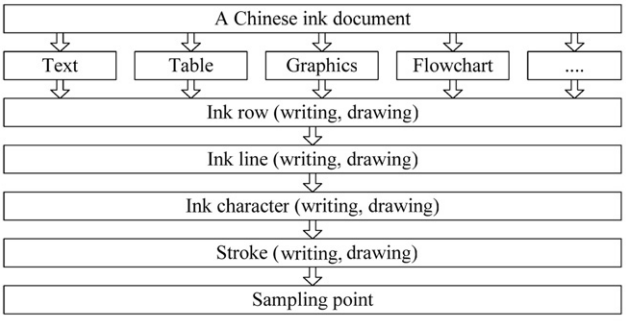
3. Modeling an ink document as a matrix

A typical Chinese document consists of text regions and non-text regions including tables, flowcharts, and graphics. A text region contains paragraphs; a paragraph contains text lines; and a text line contains Chinese characters, numbers, punctuation marks, mathematical expressions, and possibly English words. A non-text region contains simple text and graphics primitives. To uniformly represent a Chinese ink document with various objects for the purpose of extracting and segmenting text and non-text regions, we first define the following terms:

(1) *An ink character*: a set of strokes to represent a meaningful basic object, which can be a Chinese character,



(a) An ink document contains its ink rows bounded by red rectangles, ink lines bounded by blue rectangles, and ink characters bounded by green rectangles.



(b) The hierarchy of a Chinese ink document.

Fig. 1. A Chinese ink document is modeled as a hierarchy. (a) An ink document contains its ink rows bounded by red rectangles, ink lines bounded by blue rectangles, and ink characters bounded by green rectangles (b) the hierarchy of a Chinese ink document.

an English word, a number (decimal, fraction), a symbol (punctuation, mathematic, or others), or a drawing line.

(2) *An ink line*: a set of adjacent ink characters located in the same straight line, which can be a text line, a mathematical expression, a cell, or a header.
(3) *An ink row*: a set of ink lines sharing the same straight line, which can be a text line or a table row.

Thus, a Chinese ink document consists of ink rows, as shown in Fig. 1(a), and can be viewed as a matrix. We apply a Chinese ink document model shown in Fig. 1(b) to represent Chinese ink documents at multiple levels and employ a bottom-up approach for table extraction and segmentation. The model is constructed as a hierarchical structure in which each object consists of its components, making it convenient

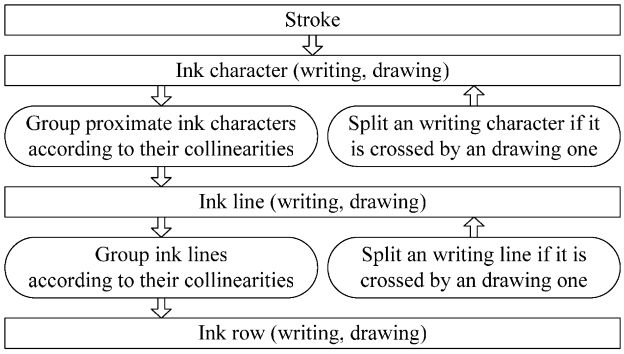


Fig. 2. A flowchart modeling a Chinese ink document as an ink matrix.

to extract information (features) at any level, downward to stroke points and upward to ink rows.

The strokes in an ink document are first grouped into ink characters according to their temporal and spatial information, and ink characters are classified as writing or drawing according to their linearity. A writing character is split if it is crossed by a drawing one. Then they are further clustered into ink lines, i.e., writing and drawing lines, in terms of their collinearities and proximities. A writing line is split if it is crossed by a drawing one. Ink lines are grouped into ink rows according to their collinearities. These entities hierarchically construct a matrix to provide more contextual information for table extraction and segmentation. Strokes are first hierarchically organized into a matrix model. The matrix model furthers the hierarchical model. It can extract tables from ink documents, segment extracted ink tables into their components (rows, columns, headers, and cells), store the extracted and segmented information, access the stored information, and transform the stored information into tables of a rational database. A flowchart describing the model for a Chinese ink document is shown in Fig. 2.

3.1. Extracting ink characters based on strokes' temporal and spatial information

Each stroke in an ink document is first set as an ink character candidate, and then the ink character candidates are merged according to their temporal and spatial information. The flowchart of extracting ink characters from a list of strokes is shown in Fig. 3.

The pen speed between ink characters is normally slower than that within an ink character. Thus, adjacent candidates in a temporal order are first grouped to form larger ones according to an adaptive speed threshold θ. This can reduce the computation load of the processing because there are fewer elements to be considered, and there is more information from multiple strokes. Each pair of adjacent points in a stroke has a pen speed $V_s(Pt_i, Pt_{i+1})$. The threshold θ is set to the minimum V_s in a group of strokes being considered as a candidate for a character. If the pen speed V_c between one candidate and the previous one is smaller than θ of the

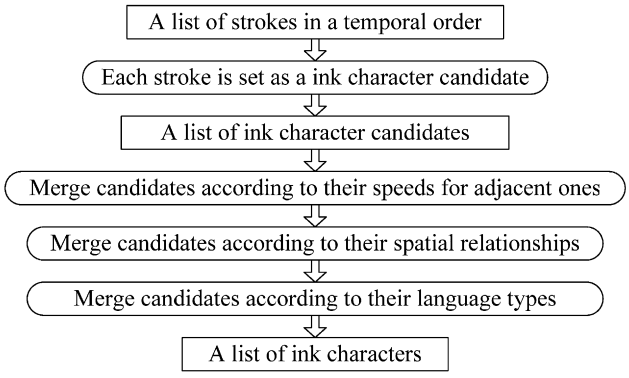


Fig. 3. A flowchart extracting ink characters from a list of strokes.

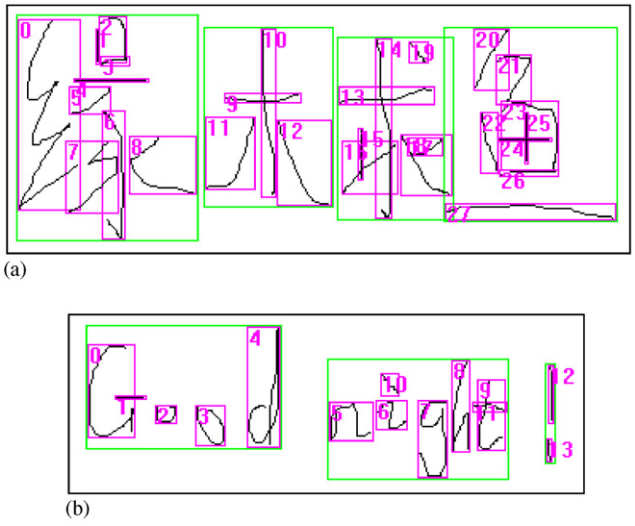


Fig. 4. Pen strokes and their orders for Chinese characters and English words. (a) Strokes 11, 12 are at the left and right of stroke 10, respectively; stroke 19 are at the top of stroke 13; strokes 15–18 are at the bottom of stroke 13; and strokes 24, 27 are at the inside and outside of stroke 23 (b) stroke 10 is at the top of stroke 6; stroke 13 is at the bottom of stroke 12; and strokes 10, 8 are at the left and right of stroke 7, respectively.

previous candidate, then they are merged. V_c is defined as $V_c = D/T$, where D is the distance between the first point of the first stroke of the current candidate and the last point of the last stroke of the previous candidate, and T is the elapsed time between the two points.

Adjacent strokes in a temporal order are not in a spatial order in a Chinese character and in an English word, i.e., the next stroke can lie at the top, bottom, left, right, inside, or outside of the current or previous one, as shown in Fig. 4, where strokes are bounded by pink rectangles and labeled with numbers to denote their writing orders. Non-adjacent character candidates in a temporal order are merged to form Chinese characters and English words according to their spatial relationships.

There is no significant gap between adjacent Chinese characters, but the width of a Chinese character is smaller than 1.5 times of its height in most cases, as shown in Fig. 4(a), where Chinese characters are bounded by green rectangles. There is a substantial gap between adjacent English words, but there is much less gap between adjacent English letters, as shown in Fig. 4(b), where English words are bounded by green rectangles. English words have different widths. Thus, ink character candidates are first merged for Chinese characters according to a width constraint, and then English words are extracted in terms of language types of candidates.

Next, ink character candidates are iteratively merged according to their enclosure, overlap and closeness relations in space. If two ink character candidates satisfy one of the following conditions, and the ratio of height/width of a merged one is smaller than 1.5, then they are merged. The latter condition is intended to avoid merging writing and drawing character candidates. If the number of ink character candidates decreases, the merging test is repeated. Otherwise, the ink character extraction terminates. The conditions for merging two ink character candidates are:

(1) The current ink character is bounded by or bounds another one.
(2) The current one overlaps another one.
(3) The current one is close to another one.

After the above process, some ink character candidates are components of wider English words because the width-to-height ratio of these English words is larger than 1.5. These components are further merged into words. We identify three kinds of stroke characteristics for ink character candidates' classification: density, orientation, and intersection.

(1) *Stroke density*: $D_C = N_C/A_C$, where N_C is the number of strokes in a candidate and A_C is the area of a candidate's bounding rectangle. The densities of English words are roughly uniform (see Fig. 4(b)), but those of Chinese characters are not. Because the stroke number of a Chinese character may vary from 1 to more than 20 and each character occupies a similar block size, the stroke density of Chinese characters varies significantly (also see Fig. 4(a)). In practice, however, many Chinese characters have very high stroke density.
(2) *Stroke orientation*: $O_C = (N_C - N_V)/N_C$, where N_V is the number of vertical strokes. If the inclination of a stroke and the x-direction ranges from 60° to 120°, the stroke is viewed as vertical. A stroke is represented by a best-fit straight line since English characters consist of mainly vertical strokes. However, many Chinese characters consist of strokes with more than one, and possibly as many as four directions.
(3) *Stroke intersection*: $I_C = N_I/N_C$, where N_I is the number of intersected stroke pairs. English characters do not contain many intersections. However, Chinese characters contain many intersections.

We combine the three characteristics of a candidate to classify a candidate as Chinese or English (or a number or

mathematical expression). If the sum of a candidate's D_C, O_C and I_C is larger than 1.0, then it is classified as Chinese; otherwise, it is classified as English. If gaps between English candidates are smaller than half of their heights, then the proximal candidates are merged into a complete English word.

It is very difficult to automatically and precisely extract correct ink characters corresponding to Chinese characters, because there are no significant gaps between Chinese characters located in the same line. A Chinese character may be mis-segmented into many ink characters because its components are far apart from each other, and many ink Chinese characters that are partially overlapped or close to each other may be mis-grouped as a character. We will correct wrong characters extracted in a later stage by exploiting more contextual information.

These steps provide us with a list of ink characters. Each ink character is classified as writing or drawing according to its linearity. Let the first and the last points of an ink character define a line L. If the maximum distance between the other points of an ink character and the line L is smaller than a threshold, then the ink character is drawing; otherwise, it is writing. The threshold is empirically determined as 8 pixels. If neighboring ink characters of a drawing character containing one stroke are writing, then they are set as writing. This is to correct misclassified drawing characters (for example, "1", or "-"). Some errors can be corrected during further processing by using more information. All strokes in a drawing character are set as drawing strokes. Each drawing stroke is split using a recursive splitting approach [29] according to a preset distance tolerance, empirically determined as 6 pixels. Thus, several bounding lines from a stroke can be obtained. Strokes in an ink document are shown in black in Fig. 1(a), where the ink characters are bounded by rectangles in green.

3.2. Extracting ink lines from a list of ink characters

Proximate ink characters sharing the same straight line form an ink line. An ink line of ink characters might include not only correctly extracted Chinese characters, but also wrong ones. They may also contain larger ink characters (e.g., Chinese characters, English words, and numbers) as well as smaller ink characters (e.g., punctuation symbols). Thus, ink characters grouped as an ink line should have a similar vertical position. Each ink character in an ink document is first set as an ink line candidate, and then the candidates are merged according to their proximation and collinearity. Thus, several strokes representing a bounding or separating line can be grouped together. An ink line is classified as writing or drawing according to the type of ink characters it contains. If two candidates simultaneously satisfy the following conditions, they are merged.

(1) Proximation: For two Chinese candidates, $D_h < \max(H_1, H_2))/2$, where D_h is the horizontal gap between two candidates, and H_1 and H_2 are their heights. For two English candidates, or a Chinese candidate and an English candidate, $D_h < (H_1 + H_2)/2$.
(2) Collinearity: $D_t < \max(H_1, H_2)/3$, and $D_b < \max(H_1, H_2)/3$, where D_t is the vertical gap of the left top points of their bounding rectangles, and D_b is the vertical gap of the left bottom points of their bounding rectangles.

In some tables, certain cells are separated by separating lines, not white spaces, and their ink characters may be very proximate to each other. Thus, they may be misgrouped as one writing line. However, some writing lines might be intersected by drawing lines. If one writing line is crossed by a drawing line, it is split by the line, and the two parts are associated. If one writing character is crossed by a drawing character, it is split by the drawing character. This can correct certain wrongly recognized writing characters and writing lines. Extracted ink lines are shown in Fig. 1(a), which are bounded by rectangles in blue.

3.3. Extracting ink rows from a list of ink lines and associating them

Ink lines sharing the same straight line can form an ink row. Each ink line is first set as an ink row candidate, and then the candidates are merged according to their collinearity relations with a similar height. If two candidates simultaneously satisfy the following conditions, they are merged.

(1) Collinearity: $D_t < \min(H_1, H_2)/3$, $D_b < \min(H_1, H_2)/3$, where D_t is the vertical gap of the left top points of their bounding rectangles, D_b is the vertical gap of the left bottom points of their bounding rectangles, and H_1 and H_2 are their heights.
(2) Height similarity: $|H_1 - H_2| < \min(H_1, H_2)/4$.

The ink rows are shown in Fig. 1(a), bounded by rectangles in red.

An ink row is classified as writing or drawing according to the type of ink lines it contains. If the horizontal or vertical gap between a writing row and its most proximate drawing row is smaller than their average height, the two rows are associated. If writing lines in vertically adjacent writing rows have approximately equal left or right edges or center positions, and/or are associated with the same vertical drawing row, then they belong to the same column. The collinear error is experimentally set to half of their average height. Partial writing lines in a writing column or row associated with the same drawing column or row belong to a sub-column or a sub-row.

3.4. Representing an ink document with a matrix

Because the writing and drawing order for table elements is not always from left to right and from top to bottom, the

extracted entities are reordered in a spatial order. Ink rows are first reordered from top to bottom. Then, ink lines in each ink row are reordered from left to right, as is each ink line of ink characters. Red order numbers of rows, blue order numbers of lines in each row, and green order numbers of characters in each line are shown in Fig. 1(a).

Ink rows form an ink matrix with multiple levels of entities. In this matrix, a higher level of entity consists of groups of entities at a lower level. The entities at each level are associated. The model facilitates table extraction because it provides not only stroke level information but also multiple levels of stroke group information. It can take contexts into account for table extraction, which is very important in our case, because tables in ink documents are often inaccurate or incomplete.

4. Table extraction and segmentation based on a matrix

A table contains rows, columns, and bounding and separating lines. A row or column contains headers and cells, and a cell may contain multiple lines of text or nothing. Therefore, a table can be modeled as a matrix. One ink row containing multiple sparse ink lines can be a table row. Adjacent ink rows with the similar distribution of ink lines form a table. After constructing a matrix of an ink document, extracting its tables requires extracting from the ink matrix a sub-matrix satisfying some constraints for a table model. Moreover, segmenting tables requires extracting table elements and their relationships.

4.1. Table extraction based on the matrix of an ink document

Tables with or without bounding and separating lines are identified by rows with identical distributions of headers and cells. Thus, we extract a table starting from a writing row with more than one writing line, and use the writing row as a seed-table. The procedure for extracting tables from an ink matrix based on seed-tables is shown in Fig. 5.

The seed-table grows based on the identical distribution of writing lines between a writing row and its adjacent ink row in the seed-table. If they have the same number of components, the following rules are used. If the writing lines with the same index of two adjacent rows overlap horizontally, they are aligned. If all writing lines of adjacent writing rows are aligned, the rows have the same distribution of writing lines.

If the vertical gap between two seed-tables is smaller than the average height of their writing rows, they are adjacent. Adjacent seed-tables with less than two ink rows are merged to group writing rows with different numbers of components but belonging to the same table. After merging seed-tables, seed-tables with less than two ink rows are identified as non-tables because, in practice, no table ever has only one row. The other seed-tables are identified as tables.

If a writing row in the ink matrix satisfies the following conditions, it is attached to a table as its caption.

(1) The gap between the writing row and the ink table is smaller than half the height of the writing row.
(2) Its first character is "表" or "Table", and its second character is a number.

The extracted ink table is shown in Fig. 6(a), which is bounded using a light blue rectangle, and whose strokes are drawn in light blue.

4.2. Table segmentation

A table visualizes indexing schemes for relations, which may be segmented as a set of n-tuples where n is the number of sets in the relation [13]. Table segmenting serves to index cells via their row and column indices.

A row or column in a table consists of headers and cells, but headers may be nested. Some tables contain no row or column headers, but only cells. Multiple writing lines sharing the same sub-row or sub-column are handled as a component because they have a common header. Row headers and cells are identified using the following rules. Likewise, column headers and cells are identified using rules based on similar principles.

(1) If the second writing line of the current writing row horizontally overlaps the first writing line of its successive writing rows, then the first writing line of the current writing row is a row header, and the second writing line of the current writing row and the first lines of the subsequent writing rows are nested row sub-headers. The other writing lines are cells.
(2) Otherwise, the first writing column may be row headers or cells, which can be identified using the character number and type. If the maximum character number of the writing lines (except the first one) of the first writing column is 60% smaller than that of the other writing columns, then the writing lines (except the first one) of the first writing column are row headers. This is to discriminate brief row headers from detailed cells. If the language types are not similar, the writing lines (except the first) of the first writing column are row headers. This is to discriminate row headers from cells with different language types.

A table can be viewed as a set of rows and columns and their nested sub-rows and sub-columns. If cells and headers have approximately equal left or right edges or center positions, and/or are associated with the same vertical drawing line if available, then they belong to a table column. Similarly, if their bottom edges are in approximately the same position, and/or they are associated with the same horizontal drawing row if available, then they belong to a table row. The collinear error is experimentally set to half of their average height.

```
i = 0, i < the number of ink rows in an ink matrix, i = i + 1
{
    If the ith ink row is a writing row, not visited, and has more than one writing line:
    {
        Set it as the current row, and generate a seed-table using it.
        (1) Get the current rows' bottom adjacent writing row.
        (2) If the adjacent ink row is a writing row, not visited, and has a associated drawing row
            in the seed-table, then add it to the seed-table, set them visited, set it as the current row,
            and go to (1).
        (3) If the adjacent ink row is a writing row, not visited, and has an identical distribution of
            writing lines with the seed-table, then add it and its associated drawing rows to the
            seed-table, set them visited, set it as the current row, and go to (1).
        (4) A seed-table is attained.
    }
}
```

Fig. 5. Tables are extracted from a matrix based on seed-tables.

The segmented ink table is shown in Fig. 6(a), where its row headers are bounded by dashed rectangles, column headers and their nested headers by dotted rectangles, and cells by solid rectangles in pink. Headers and their nested headers are connected by their left bottom points in lines. Rows and columns are bounded by their enclosing rectangles in green and blue, respectively.

A cell or header in a segmented ink table can be accessed via its row and column indices. The extracted information is stored in a nested matrix which can handle index headers, cells, rows, and columns.

4.3. Modification and conversion of ink tables

Segmented ink tables can be modified with desired styles, e.g., each column of cells is aligned at the left, center, or right, and each row of cells is aligned at the bottom, center or top. Fig. 6(b) shows a modified version of the ink table shown in Fig. 6(a), where each row of elements is aligned at the bottom with the same inter-row gap, each column of elements is aligned at the left with the same inter-column gap, and each line of characters is aligned at the bottom.

The segmented table has thus been recognized, as shown in Fig. 6(b). Some wrongly extracted Chinese characters and numbers are corrected manually, as are wrongly recognized ones. Then the recognized table is converted into a table document for formal representation, and further converted into a spreadsheet file for statistics, as shown in Fig. 6(c) and (d), respectively. A handwriting recognizer such as Microsoft Tablet PC Platform SDK [30] was used. Tables in rational databases can be abstracted as a matrix. So the extracted and recognized information from an ink table can be easily transformed as a table of a rational database.

5. Experimental results and performance analyses

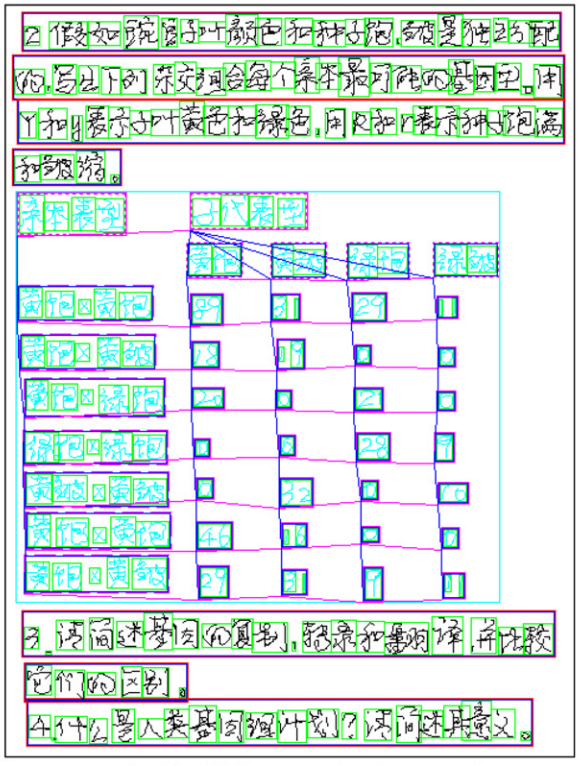
Based on the proposed approach, a software prototype has been developed in Visual C++ R7.0. This section presents performance evaluation and comparison based on the experimental results on real-life ink documents containing tables, and gives quantitative analysis based on ground-truth data.

5.1. Experimental results

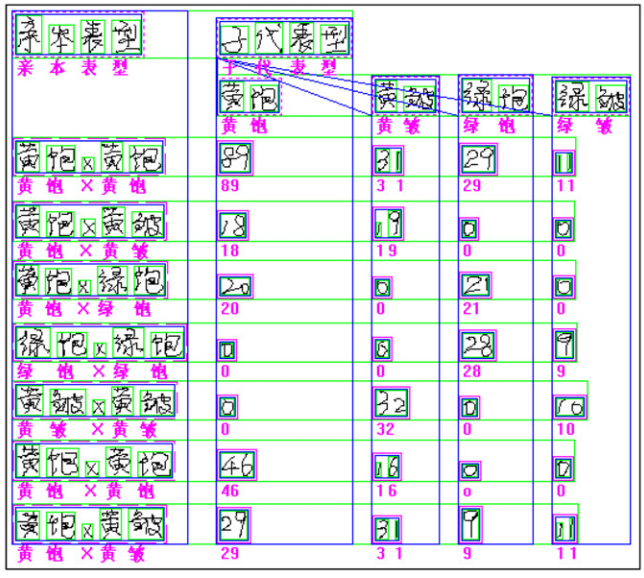
To evaluate the performance of the proposed method, we experimented with 30 Chinese ink documents containing 150 pages and 50 tables on our software prototype. These test tables cover the instances mentioned in Section 1. The documents were collected from many different sources without any restriction on the style or content of data, which includes text, graphics, flow charts, tables, and so on. The tables can be drawn at random, and may be large or small, skewed or not aligned. The Anoto Pen [31], from Hitachi Maxell Corporation Ltd., Japan, is used to handwrite the Chinese documents on Anoto papers, and the pen captures the point coordinates and timestamps for each stroke.

The experimental results are shown in Fig. 7 to illustrate the effectiveness of our approach. It is clear that many kinds of tables are extracted and segmented correctly.

Liang et al. [32] present quantitative performance measures on document structure extraction algorithms in terms of the rates of correct, missing, false, merging, splitting, and spurious detections. The performance evaluation of our approach is done at the table level for table extraction and at the header and cell level for table segmentation; i.e., it evaluates the proportion of strokes, captions, headers, cells, rows, and columns of tables correctly extracted. The precision rate



(a) An ink table is extracted and segmented.



(b) It is modified and recognized.

亲本表型	子代表型			
	黄饱	黄皱	绿饱	绿皱
黄饱 X 黄饱	89	31	29	11
黄饱 X 黄皱	18	19	0	0
黄饱 X 绿饱	20	0	21	0
绿饱 X 绿饱	0	6	28	9
黄皱 X 黄皱	0	32	0	10
黄饱 X 黄饱	46	16	0	0
黄饱 X 黄皱	29	31	9	11

(c) The recognized table is converted into a table document.

	A	B	C	D	E	F
1	亲本表型	子代表型	子代表型	子代表型	子代表型	
2		黄饱	黄皱	绿饱	绿皱	
3	黄饱X黄饱	89	31	29	11	
4	黄饱X黄皱	18	19	0	0	
5	黄饱X绿饱	20	0	21	0	
6	绿饱X绿饱	0	6	28	9	
7	黄皱X黄皱	0	32	0	10	
8	黄饱X黄饱	46	16	0	0	
9	黄饱X黄皱	29	31	9	11	
10						

(d) The recognized table is imported into a spreadsheet application.

Fig. 6. An ink table is extracted, segmented, modified, recognized, and converted. (a) An ink table is extracted and segmented; (b) it is modified and recognized; (c) the recognized table is converted into a table document; (d) the recognized table is imported into a spreadsheet application.

ρ and the recall rate q are used to measure the average performance of detected objects: $\rho = \eta/(\eta+\varepsilon)$, $q = \eta/(\eta+\vartheta)$, where η is the number of true positive objects that are detected correctly, ε is the number of true negative objects that are detected incorrectly, and ϑ is the number of positive objects that are mis-detected [33].

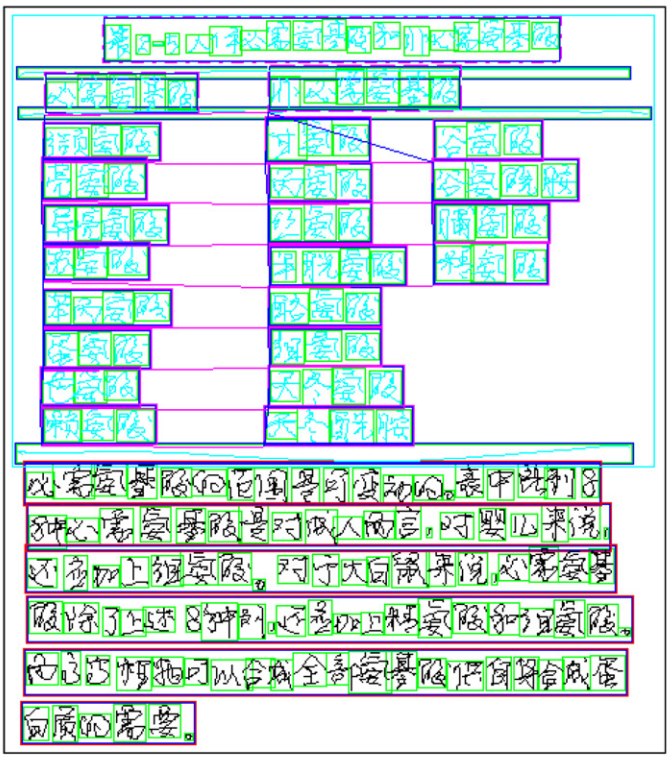
To evaluate the extracted results quantitatively, the ground-truth data are used as a reference. As the reference data cannot be obtained by any automatic processing method, they are determined manually by professional engineers. The average performance of our algorithm in table extraction and segmentation is given in Table 1.

The processing time is another important performance index. The processing times of extracting and segmenting tables in our ink documents were tested on a PC with a PIII 1.6 GHz CPU and 512 M RAM, where the maximum processing time was 0.57 s, and the average time 0.35 s.

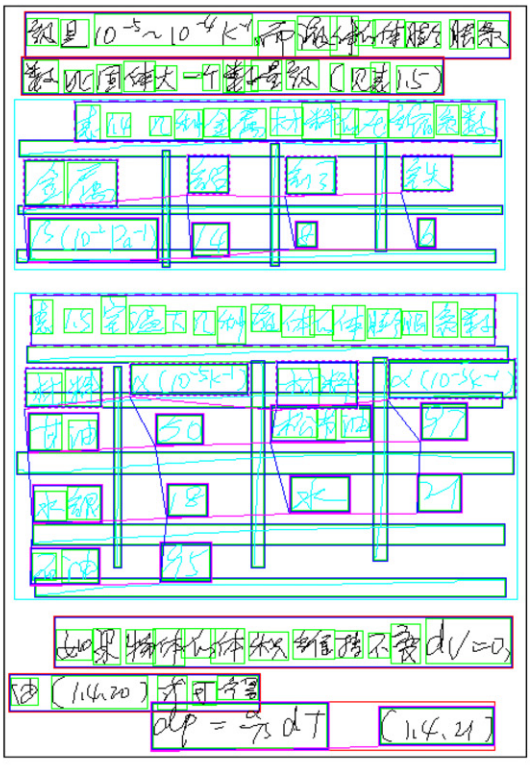
5.2. Comparison with related work

There has been some investigation into the segmentation of ink documents, but only Jain et al. [12] extract and segment ink tables. Although their approach is only for English documents and is not based on the same test data, it is the most comparable work with ours. We compare their approach and ours in four aspects: table extraction, table segmentation, result accessibility, and computational cost. The comparison results are listed in Table 2.

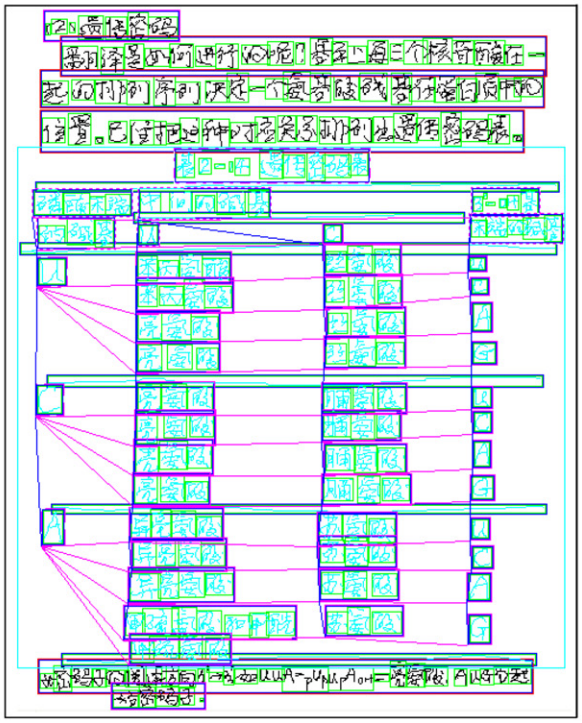
In order to compare the performance of the approach of Jain et al. [12] with ours, we implemented their approach in



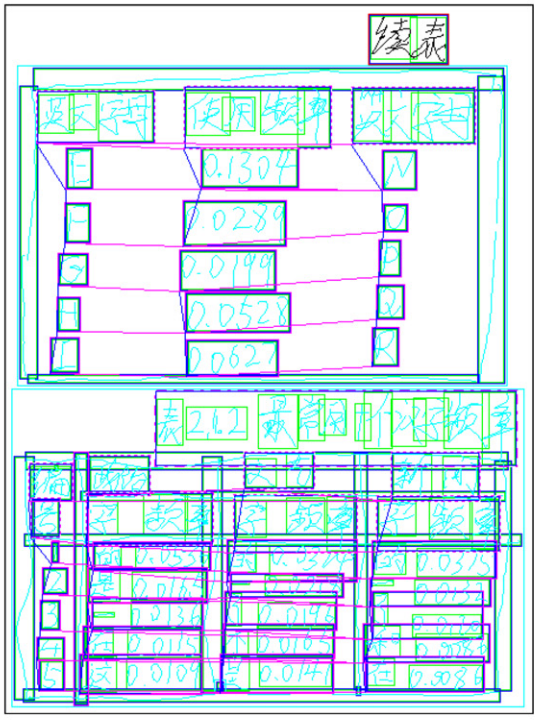
(a) An ink table containing two border lines and one separating line.



(b) An ink table containing numbers, complete separating lines, and incomplete bounding lines.



(c) An ink table containing nested headers but not full length separating lines.



(d) An ink table containing Chinese characters, numbers, and complete bounding lines.

Fig. 7. Ink tables are extracted and segmented. (a) An ink table containing two border lines and one separating line; (b) an ink table containing numbers, complete separating lines, and incomplete bounding lines; (c) an ink table containing nested headers but not full length separating lines; (d) an ink table containing Chinese characters, numbers, and complete bounding lines.

Table 1
Average performance of our approach

	Correctly detected	Incorrectly detected	Mis-detected	Precision (%)	Recall (%)
Stroke	15 592	258	286	98.4	98.2
Caption	41	2	4	95.3	91.1
Header	303	11	23	96.5	92.9
Cell	1121	84	75	93.0	93.7
Row	398	17	18	95.9	95.7
Column	202	21	22	90.6	90.2

Table 2
Comparison between two approaches

Characteristic	Approach	Hierarchical approach [12]	Our approach
Table extraction	No bounding lines and with separating lines	No	Complete
	No separating lines and with bounding lines	No	Complete
	Nested headers and cells	Partial	Complete
	Headers and cells	Complete	Complete
Table segmentation	Nested headers and cells	Partial	Complete
	Association between table elements	No	Yes
Result accessibility		High	High
Computational cost		High	Moderate

Table 3
Average performance of the approach of Jain et al. [12]

	Correctly detected	Incorrectly detected	Mis-detected	Precision (%)	Recall (%)
Stroke	14 472	978	1406	93.7	91.1
Header and cell	1224	345	298	78.0	80.4
Row	356	46	70	88.6	83.6
Column	186	34	38	84.5	83.0

software. It neither extracts captions for tables, nor discriminates headers from cells for table segmentation. Its average performance on our test data is shown in Table 3.

Comparing Tables 1 with 3, we can see that our approach has higher precision and recall rates. This is because we consider nested headers and depend less on bounding and separating lines, but instead utilize the distributional characteristics of headers and cells as well as the association between writing and drawing entities.

5.3. Error analyses

Failure cases in our extraction are illustrated in Fig. 8, including extraction of tables, their captions, headers, cells, rows, and columns. These errors are analyzed as follows. Note that most of the errors can be handled by fusing semantic information in the future.

(1) Extraction of tables. When adjacent lines of texts have sparse components, they are mistaken for components of tables. In other words, a two-column (or multi-column) ink document in more than one line will be identified as a table.

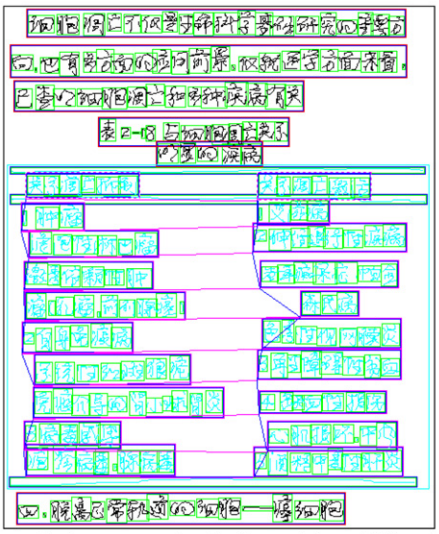
(2) Extraction of captions. An error occurs when a caption contains multiple lines of text, causing incorrect recognition of its first and second characters.

(3) Extraction of headers and cells. Most errors occur when a header or cell contains multiple lines of text. It is difficult to identify these lines of text as belonging to a header (cell) or multiple headers (cells) when there are no separating lines because some cells contain nothing. If a header or cell contains more than one English word or sparse Chinese characters, the words or characters may be identified as multiple headers or/and cells. Furthermore, if some nested headers are at the center of a shared header, and they are not grouped by a separating line, then they may be wrongly identified.

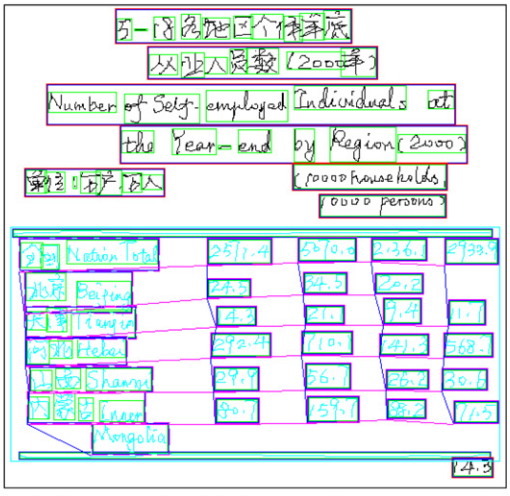
(4) Extraction of rows and columns. Some wrongly extracted headers and cells can result in the wrong extraction of rows and columns.

5.4. Discussions

From the above experimental results and our performance analyses, it can be concluded that the proposed approach



(a) The caption is not extracted, and cells containing multi-line of text are not extracted correctly, because they are separated by digits, not lines.



(b) The caption is not extracted, and the last two headers are not extracted correctly, because it is a header containing multi-line of text.

Fig. 8. Failure cases of extracted and segmented tables. (a) The caption is not extracted, and cells containing multi-line of text are not extracted correctly, because they are separated by digits, not lines (b) the caption is not extracted, and the last two headers are not extracted correctly, because it is a header containing multi-line of text.

has four major advantages:

(1) It uses a matrix model, which can provide more contextual support and evidence for table extraction.
(2) More sources of information are exploited, including components of rows, bounding and separating lines, and their combination, to improve the extraction of tables.
(3) Association between writing rows and corresponding drawing ones improves the segmentation of tables.
(4) A segmented ink table is represented as a nested matrix, which not only stores all the information extracted and segmented, but also supports the access of headers and cells using row and column indexes.

Consequently, the proposed approach is able to achieve satisfactory results in the task of extracting tables, their elements and relationships among extracted elements.

6. Conclusions

This paper has proposed a matrix model for extracting and segmenting ink tables. Ink tables with or without bounding and separating lines, as well as those with nested headers, can be processed, since we focus on the distribution of headers and cells. Extracted ink tables are decomposed into nested headers and cells based on the association between rows and columns as well as bounding and separating lines.

The proposed approach and its software have been tested using many ink documents containing ink tables. Their performance analyses are reported here, including the test results and comparative evaluation relative to another published method. The analyses confirm that the proposed approach is more effective and robust than other approaches currently available.

Acknowledgments

The work described in this paper was substantially supported by five projects: the National Key Basic Research and Development Program of PR China (Grant no. 2002CB312103, 2006CB303105), the National Natural Science Foundation of PR China (Grant no. 60605018, 600373056), and the Hong Kong RGC project (Grant no. CUHK4205/04E).

References

[1] Anoto AB, ⟨http://www.anoto.com/⟩.
[2] Microsoft Windows XP Tablet PC Edition 2005, ⟨http://www.microsoft.com/windowsxp/tabletpc/default.mspx⟩.
[3] IBM ThinkPad TransNote, ⟨http://www.research.ibm.com/electricInk/⟩.
[4] Wacom's Cintiq, ⟨http://www.wacom.com/lcdtablets/index.cfm⟩.
[5] SMART Board, ⟨http://www.smarttech.com⟩.
[6] A. Meyer, Pen computing: a technology overview and a vision, SIGCHI Bull. 27 (3) (1995) 46–90.
[7] R. Davis, J. Landay, T. Stahovich, R. Miller, E. Saund, Making pen-based interaction intelligent and natural, AAAI Fall Symposium, October 21–24, 2002–2004, Arlington, Virginia, Technical Report FS-04-06: 174.
[8] Microsoft Windows XP Tablet PC Edition Software Development Kit 1.7, ⟨http://www.microsoft.com/downloads/details.aspx?familyid = b46d4b83-a821-40bc-aa85-c9ee3d6e9699&displaylang = en⟩.
[9] IBM Ink Manager SDK for ThinkPad TransNote, ⟨http://www-306.ibm.com/software/voice/viavoice/dev/transnote_sdk.html⟩.
[10] Y. Wang, I.T. Phillips, R.M. Haralick, Table structure understanding and its performance evaluation, Pattern Recognition 37 (7) (2004) 1479–1497.
[11] S. Lewandowksy, I. Spence, The perception of statistical graphs, Sociol. Methods Res. 18 (2 and 3) (1989) 200–242.
[12] A.K. Jain, A.M. Namboodiri, J. Subrahmonia, Structure in on-line documents, Proceedings of the Sixth International Conference on Document Analyses and Recognition, 2001, pp. 844–848.

[13] R. Zanibbi, D. Blostein, J.R. Cordy, A survey of table recognition: models, observations, transformations, and inferences, Int. J. Document Analyses Recognition 7 (1) (2004) 1–16.
[14] E.H. Ratzlaff, Inter-line distance estimation and text line extraction for unconstrained online handwriting, Workshop on Frontiers in Handwriting Recognition (2000) 33–42.
[15] R. Bozinovic, S. Srihari, Off-line cursive script word recognition, IEEE Trans. Pattern Anal. Mach. Intell. 11 (10) (1989) 68–83.
[16] L.D. Ernest, Machine recognition of cursive script, Proceedings of the IFIP Congress, vol. 62, 1992, pp. 462–466.
[17] M. Shilman, Z. Wei, S. Raghupathy, P. Simard, D. Jones, Discerning structure from freeform handwritten notes, Proceedings of the Sixth International Conference on Document Analyses and Recognition, vol. 1, 2003, pp. 60–65.
[18] J. Blanchard, T. Artières, On-line handwritten documents segmentation, in: F. Kimura, H. Fujisawa (Eds.), Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition, Tokyo, Japan, 2004, pp. 148–153.
[19] H. Chen, G. Loudon, Y. Wu, R. Zitserman, Segmentation and recognition of continuous handwriting Chinese text, Int. J. Pattern Recognition Artif. Intell. 12 (2) (1998) 223–232.
[20] C.M. Bishop, M. Svensen, G.E. Hinton, Distinguishing text from graphics in on-line handwritten ink, in: F. Kimura, H. Fujisawa (Eds.), Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition, Tokyo, Japan, 2004, pp. 142–147.
[21] G. Nagy, Twenty years of document image analyses in PAMI, IEEE Transactions on Pattern Anal. Mach. Intell. 22 (1) (2000) 38–62.
[22] K.-H. Lee, Y.-C. Choy, S.-B. Cho, Geometric structure analyses of document, images: a knowledge-based approach, IEEE Trans. Pattern Anal. Mach. Intell. 22 (11) (2000) 1224–1240.
[23] A.K. Jain, B. Yu, Document representation and its application to page decomposition, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) 294–308.
[24] T. Watanabe, Q. Luo, N. Sugie, Layout recognition of multi-kinds of table-form documents, IEEE Trans. Pattern Anal. Mach. Intell. 17 (4) (1995) 432–445.
[25] J. Hu, R. Kashi, D. Lopresti, G. Wilfong, Table structure recognition and its evaluation, SPIE Document Recognition and Retrieval VIII, San Jose, CA, January 2001.
[26] H.T. Ng, C.Y. Lim, J.L. Koo, Learning to recognition tables in free text, Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, 1999, pp. 443–550.
[27] P. Pyreddy, W.B. Croft, Tintin: a system for retrieval in text tables, Proceedings of the Second International Conference on Digital Libraries, 1997, pp. 193–200.
[28] Y. Wang, J. Hu, Detecting tables in HTML documents, Lecture Notes on Computer Science, vol. 2423, Springer, Berlin, 2002, pp. 249–260.
[29] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analyses, and Machine Vision, second ed., Brooks/Cole a division of Thomson Asia Pte Ltd., United States of America, 1998 p. 243.
[30] Windows XP Tablet PC Edition 2005 Recognizer Pack, ⟨http://www.microsoft.com/downloads/details.aspx?displaylang = zh-cn&FamilyID = 080184DD-5E92-4464-B907-10762E9F918B⟩.
[31] Hitachi Maxell Corporation, Ltd., Japanese, ⟨http://www.maxell.co.jp/e/products/industrial/digitalpen/products.html⟩.
[32] J. Liang, I.T. Phillips, R.M. Haralick, Performance evaluation of document structure extraction algorithms, Comput. Vis. Image Understanding 84 (1) (2001) 144–159.
[33] J. Fan, Y. Gao, H. Luo, G. Xu, Statistical modeling and conceptualization of natural images, Pattern Recognition 38 (6) (2005) 865–885.

About the Author—XI-WEN ZHANG is a Postdoctoral Fellow in the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China. He received a Ph.D. in Mechanical Manufacturing and Automation from Dalian University of Technology, Dalian, China, in 2000. His research interests include human computer interaction and ink document understanding.

About the Author—MICHAEL R. LYU is currently a Professor in the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China. He is elected to IEEE Fellow in 2004. He received a Ph.D. degree in Computer Science from University of California, Los Angeles, USA, in 1988. Dr. Lyu's research interests include software reliability engineering and video searching and delivery.

About the Author—GUO-ZHONG DAI is a research professor in Institute of Software, The Chinese Academy of Sciences, Beijing, China. He graduated from the Department of Application Mathematics, University of Science and Technology of China, Beijing, China, in 1967. His major research interest covers Human Computer Interaction and Software Engineering.