Robust Cell Extraction Method for Form Documents based on Intersection Searching and Global Optimization

Hiroshi Tanaka, Hiroaki Takebe and Yoshinobu Hotta
Fujitsu Laboratories Ltd.
4-1-1 Kamikodanaka, Nakahara-ku, Kawasaki 211-8588, Japan
e-mail: htnk@jp.fujitsu.com

Abstract— **This paper presents a method to extract cells from form documents of an unknown format based on intersection tracking and global optimization. Because of the increasing varieties of forms and complications in the forms, cell extraction has become a key factor in automatic form recognition. The proposed method produces cell candidates by using intersection features and tracing intersection points to form closed regions. Then, global optimization is carried out to determine a set of cells having the highest likelihood. Maintaining precision for 82 kinds of forms, the proposed method achieved a recall rate of 80.4% whereas the conventional method achieved a recall rate of 77.1%.**

Keywords: form recognition, cell extraction, OCR, intersection feature, ruled line, dynamic programming

I. INTRODUCTION

One of the main objectives of form image recognition is to recognize text data written in the forms, thereby reducing data entry costs of the operators. A high-performance form recognition system can reduce the operator's burden; however, low-performance system can lead to an increase in the cost. Most of the practical form recognition systems use form templates to yield highly accurate results. To deal with various types of forms, some form recognition systems adopt form identification method [1]-[4]. However, there is an increasing demand for template-free form recognition methods [5]-[7], which is backed by the growing market of scanners and digital cameras (note: [7] is not a method of table image recognition). As the use of scanners is gaining popularity primarily in the consumer market, the nature of document images that require to be recognized is becoming more complicated and degraded in quality. Form recognition can be applied to even documents that are not forms but which include tables; these form recognition methods primarily consist of table recognition modules.

Tables can be of two types: ruled-line and a non ruled-line tables. A ruled-line table is formed by ruled lines that are straight vertical and horizontal lines, and the areas enclosed by these ruled lines are called cells, which contains text. A non ruled-line table uses a relative position between text areas to express table structures. Recognizing ruled-line tables is often considered to be easier than recognizing non ruled-line tables because ruled-line tables provide more information than non ruled-line tables. However, ruled-line tables can be difficult to recognize in the following case. In much degraded images, ruled lines are not clear enough to be extracted without errors. If some of the ruled lines are lost, the cells formed by the lost ruled lines are dropped, and the text in those cells will not be recognized correctly. As ruled line and text images may affect each other [8], extracting ruled lines and text from ruled-line tables can sometimes be more difficult than extracting text from non ruled-line tables.

Accurately extracting cells from ambiguous ruled-line images is the main topic of this paper. In earlier studies, we identified three types of cell extraction methods: direct extraction methods, ruled-line-based methods, and intersection-based methods. Direct extraction methods identify rectangular areas using geometric hashing [9] or a generalized Hough transform [10]. Ruled-line-based methods use extracted ruled lines to identify enclosed areas. A typical ruled-line-based method recursively cuts a table area using parallel ruled lines [1][11] (Fig.1). Intersection-based methods extract intersections and identify rectangles on the basis of sets of intersections [12]. Most of these methods extract only rectangles as cell areas. This indicates that they are not robust against misrecognition of ruled lines. For example, non rectangle (L-shaped) cells, as shown in Figure 2, cannot be extracted, and improper ruled lines can produce unexpected L-shaped cells.

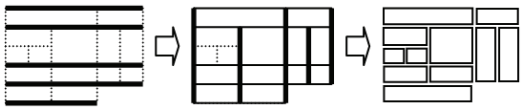

Figure 1. Cell Extraction by Parallel Ruled Lines.

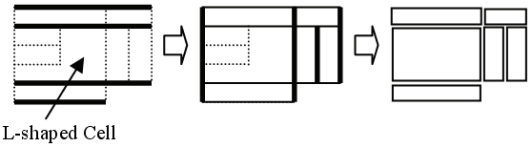

Figure 2. Misextraction of L-shaped Cell.

In this paper, we present a cell extraction method using intersection tracking and global optimization of multiple cell combinations. As intersection tracking is robust against misrecognition of ruled lines and can extract non rectangular enclosed areas, our method can achieve a high accuracy in extracting cells. In addition, we adopt a global optimization approach based on dynamic programming to obtain the best cell combination derived from multiple cell candidates.

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.79

In Section II, we describe the cell candidate extraction method that extracts intersections and identifies cell areas by tracking the extracted intersections. In Section III, we present the global optimization method, which obtains the best cell combination. In Section IV, the effectiveness of the developed method is shown through experimental results. Finally, in Section V, we conclude the paper.

II. CELL CANDIDATE EXTRACTION

In this section we describe the cell candidate extraction step that is composed of two substeps: extraction of intersections and identification of cell areas. To extract intersections, a grid space, which expresses simplified positions of ruled lines, is created. The intersections are placed on grid points and cell candidates are extracted by tracking the intersections along the grid lines.

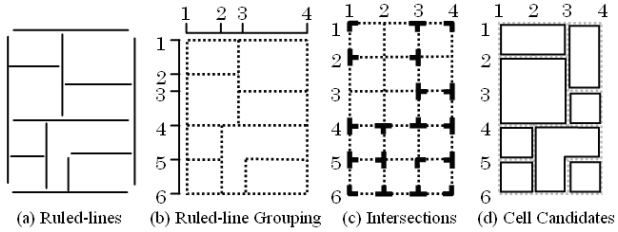

Figure 3. Steps of Cell Candidate Extraction.

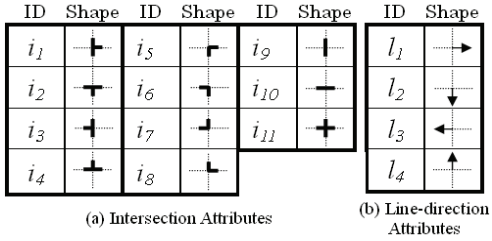

Figure 4. Intersection and Line-direction Attributes.

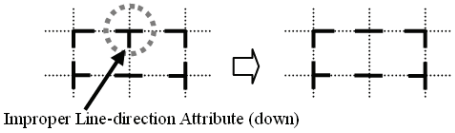

Figure 5. Deletion of Improper Line-direction Attributes.

A. Extracting Intersections

Input ruled lines are first separated into groups having similar positions (referring to the x-values for vertical lines and the y-values for horizontal lines). In Figure 3, there are six groups of horizontal lines and four groups of vertical lines (Fig.3 (b)). The intersections are extracted and assigned grid points that indicate the relations between the ruled lines and the grid positions (Fig.3 (c)).

Figure 4 shows the various types of intersections. Intersection attributes (Fig.4(a)) express the shapes of intersections ("T", "L", "I" and "+" shapes) depending on how the ruled lines are connected to the grid point. A ruled line can be connected to a grid point from four directions that are called line-direction attributes (Fig.4(b)). Each intersection attribute is composed of a combination of line-direction attributes. After the intersections are assigned, some of the incorrect conditions should be corrected by removing improper line-direction attributes (Fig.5).

B. Identifying Cell Areas by Tracking Intersections

A cell area can be extracted as a closed path formed by intersection tracking starting from the top-left corner. From each intersection, a tracking path moves the current position clockwise to the next grid point. If the path direction (Fig.6) assigned to the grid point matches the direction of the tracking path movement, the tracking path changes its path direction at the grid point. Finally, if the tracking path comes back to the starting point, the tracking is complete, and the closed path is recognized as a cell area. Figure 7 shows an example of cell candidate extraction. The dotted circles indicate the starting positions of tracking paths and are considered as representative points of each cell area.

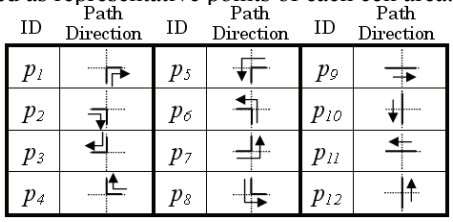

Figure 6. Path-direction Attributes.

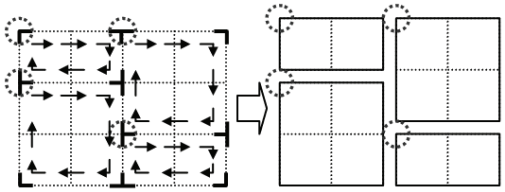

Figure 7. Cell Candidate Creation.

III. OPTIMIZATION OF CELL COMBINATIONS

If all ruled lines were extracted correctly, accurate cell extraction can be achieved using the method described in Section II. However, in reality, some of the ruled lines may be extracted incorrectly, and this may lead to cell extraction errors. To achieve accurate cell extraction with ambiguous ruled line information, we adopt a global optimization method using multiple cell candidates.

A. Combining Multiple Cell Candidates

In the case that ruled line information is not reliable, line-direction attributes can be ambiguous, as shown in Figures 8(a) and (b). Here, it is difficult to determine whether the vertical ruled line is connected to the horizontal ruled line or not. Thus, there could be two possible tracking paths, as shown in Figure 8(c) and (d). Similarly, we can obtain multiple cell candidates at each grid point.

Then, we determine an appropriate combination of multiple cell candidates. This problem can be formulated as a panel layout problem (Fig.9). To obtain an optimal cell combination, we adopt probability measure as an evaluation function. When the probability of a cell X_i is $P(X_i)$, the

evaluation function of a cell candidate set $\boldsymbol{X} = X_1,X_2,...,X_N$ can be defined by $P(\boldsymbol{X})$ as given in Formula 1. The probability measure is usually used in the form of a log likelihood as shown in Formula 2.

$$P(X) = \prod_{i=1}^{N} P(X_i) \tag{1}$$

$$\begin{aligned} L(X) &= \log(P(X)) \\ &= \sum_{i=1}^{N} \log(P(X_i)) \\ &= \sum_{i=1}^{N} Lc(X_i) \end{aligned} \tag{2}$$

Although Formula 2 can be used as an evaluation function when comparing cell combinations having the same number of cells, it needs an additional correction factor to make it applicable to a number of cell candidates. In similar problems, such as evaluating optimal candidates in language processing, Formula 3 is often used. In Formula 3, p is a predefined correction factor to compensate for a large value of N. On the other hand, Formula 4 is also used to compensate for the number of cell candidates. We adopt Formula 4 because it does not use predefined parameters and does not need specific adjustment.

$$L(X) = \sum_{i=1}^{N} Lc(X_i) + Np \tag{3}$$

$$L(X) = \frac{1}{N} \sum_{i=1}^{N} Lc(X_i) \tag{4}$$

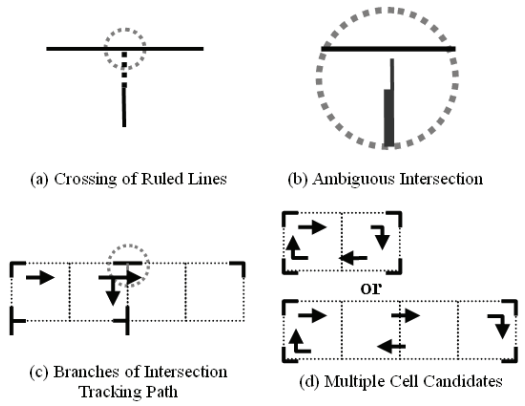

Figure 8. Multiple Cell Candidate Creation.

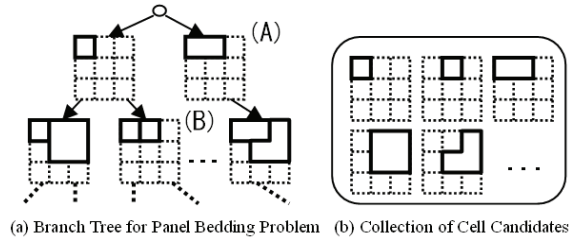

Figure 9. Combinatorial Searching for Cell Extraction.

B. Assigning Likelihood to each Cell Candidate

We have approximately defined how to estimate the log likelihood of each cell candidate $Lc(X_i)$ as follows. The log likelihood of a cell candidate can be calculated using line-direction attributes because they represent how reliable ruled lines and cell candidates are. A cell candidate is composed of a set of intersections. These intersections can be expressed as a sequence of path directions. Because a path direction is a combination of line directions, the log likelihood of a cell candidate can be calculated from the scores of the line directions.

We have defined the line-direction score in Formula 5; where the gap between a ruled line and a grid point is represented as d (Fig.10(a)). The path-direction scores (Fig.10(c)) are defined in Formula 6. The cell candidate's log likelihood is calculated using Formula 7 and 8; in these formulas the number of intersections on the intersection tracking path is represented as M.

$$\begin{aligned} &if\,(d > L/3) \qquad Sp = 0.0 \\ &else \quad if\,(d > 0) \qquad Sp = 1 - 3 \times d / L \\ &else \qquad Sp = 1.0 \end{aligned} \tag{5}$$

$$\begin{aligned} Se(rt) &= Sp(d) \times Sp(r) \\ Se(st) &= Sp(d) \times Sp(u) \times \{1 - Sp(r)\} \\ Se(lt) &= Sp(d) \times Sp(l) \times \{1 - Sp(r)\} \times \{1 - Sp(u)\} \end{aligned} \tag{6}$$

$$Le(i,j,m) = \frac{Se(m)}{Se(rt) + Se(st) + Se(lt)} \tag{7}$$

$$Lc(i,j) = \frac{100}{M} \times \sum_{k}^{M} Le(i,j,m_k) \tag{8}$$

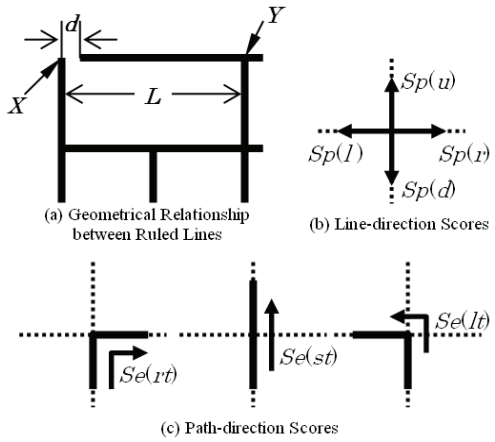

Figure 10. Scores of Line Directions and Path Directions.

C. Searching an Optimal Cell Combination by Dynamic Programming

The best combination of cell candidates can be obtained by using an optimization algorithm. We solved this problem by DP (dynamic programming). A problem that can be

Figure 11. Examples of Cell Combination for each Line.

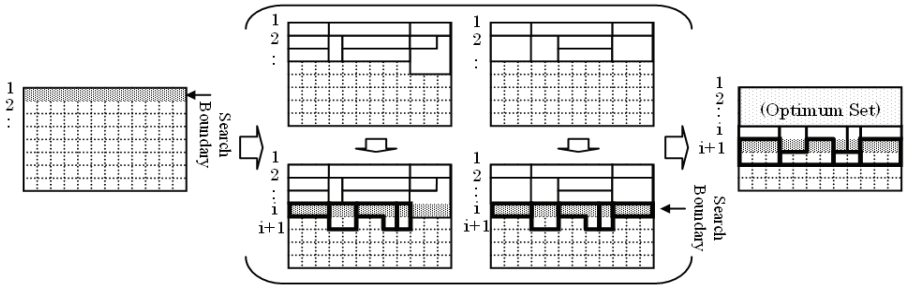

Figure 12. Procedure of Combinatorial Search of Cell Candidates

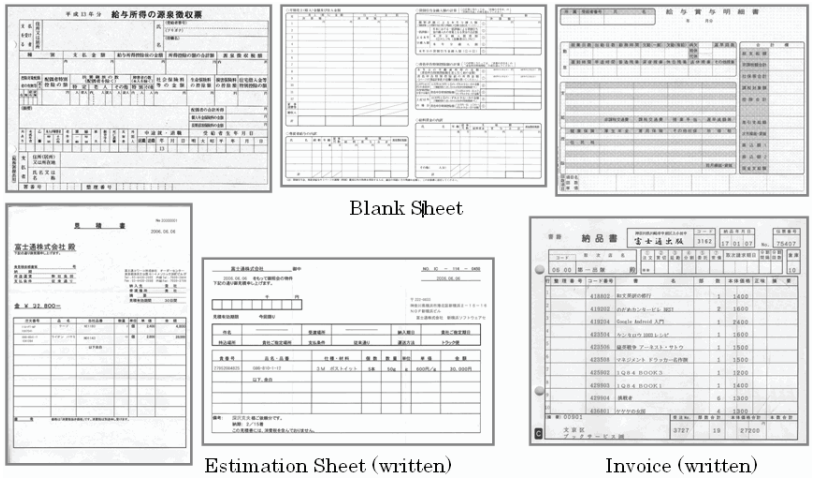

Figure 13. Examples of Form Image

solved by DP should be a multistage decision process that satisfies the following conditions.

1) The i+1 th state $s(i+1)$ can only be defined by $s(i)$ and i+1 th operation $d(i+1)$. (independent from the states $s(i-1)$ or earlier)
2) The value of the total evaluation function f is defined by a sum of all the evaluation values of states and operations.

To satisfy condition 1), we consider all possible combinations on each line as states of the DP algorithm (Fig.11). The current line filled with cell combinations is called the search boundary. This boundary is placed at the 1st line at the first step, and it is progressively moved to the bottom, as shown in Figure 12. If states on the same search boundary have the same bottom shape, their scores are compared, and only the state having the best score is selected. Finally, at the bottom of the table, the best cell candidate combination can be obtained.

IV. EVALUATION

We evaluated our method using three types of form document images. Set 1 consists of 50 images of fill-in forms without post-printed text (blank) containing 4693 cells, set 2 consists of 15 images of estimate sheets with written text (written) containing 1041 cells, and set 3 consists of 17 images of invoices (written) containing 2446 cells.

Table I shows the results of cell extraction along with the recall and precision rates. In this evaluation, we used a cell extraction method using the conventional parallel ruled-lines method (Fig.1). The results of the proposed method are relatively more accurate than those of the conventional method. It is observed that the proposed method is effective especially for images that are difficult to recognize. Figure 14 shows an example of the cell extraction results for the image containing an L-shape cell.

Table II show the results in which correct ruled lines are used to extract cells. Although we expected the results to be perfect (100 %), some of the cells could not be extracted. Figure 15 shows one of the cases in which a cell was not extracted even when correct ruled lines were input. The number of intersections in our study was limited to 10 or lower, to suppress unnatural cells from being extracted; however, in this case, the cell was not extracted because the tracking path passed 12 intersections.

TABLE I. ACCURACY OF CELL EXTRACTION

	Conventional		Proposed	
	Recall	Precision	Recall	Precision
Fill-in form (blank)	75.7%	81.9%	80.0%	81.7%
Estimate sheet (written)	87.1%	93.8%	87.9%	92.4%
Invoice (written)	71.9%	77.6%	75.3%	79.0%
Total	77.1%	83.3%	80.4%	83.1%

TABLE II. ACCURACY OF CELL EXTRACTION (WITH CORRECT RULED LINES)

	Recall	Precision
Fill-in form (blank)	99.0%	99.8%
Estimate sheet (written)	99.0%	99.1%
Written invoice (written)	99.1%	99.0%
total	99.0%	99.5%

(a) Input Image (b) Result of Conventional Method (c) Result of Proposed Method

Figure 14. Examples of L-shaped Cells.

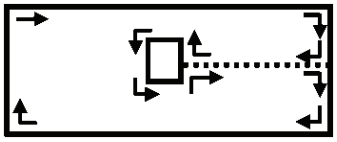

Figure 15. Case of Cell Dropping Out.

V. CONCLUSION

In this paper, we have described a cell extraction method that is effective even when the input ruled lines are unclear. This method consists of two substeps; (1) the cell candidate extraction method that uses intersection tracking and (2) the global optimization method that can obtain the optimal cell combination from multiple cell candidates using the DP algorithm. By using intersections which is more reliable than directly using ruled lines, our method achieved a higher accuracy than the conventional method used in this study, particularly, in low-quality images.

As the next step, it is necessary to improve our method to achieve greater accuracy for extracting complex shape cells such as the one shown in Figure 15. In addition, we will improve the method for estimating the cell candidate's log likelihood. To achieve the goals, we should use larger standard databases to evaluate the effectiveness of our method. This study is just the first step in perfecting cell extraction. We believe that further study of log likelihood estimation will make our method more practical.

REFERENCES

[1] T. Watanabe, Q. Luo, and N. Sugie, "Toward a Practical Document Understanding of Table-form Documents: Its Framework and Knowledge Representation," Proc. 2nd ICDAR, pp. 510-515, Oct. 1993

[2] S. W. Lam, L. Javanbakht, and S. N. Srihari, "Anatomy of a FormReader," Proc. 2nd ICDAR, pp. 506-509, Oct. 1993.

[3] K.-C.Fan, Y.-K.Wang, and M-L.Chang, "Form Document Identification Using Line Structure Based Features," Proc. 6th ICDAR, pp. 704-708, Sept. 2001.

[4] L. A. D. Hutchison, and W. A. Barrett, "Fast Registration of Tabular Document Images Using the Fourier-Mellin Transform," Proc. 1st DIAL, pp. 253-267, Jan. 2004.

[5] D.W.Embley, D.Lopresti, and G.Nagy, "Notes on Contemporary Table Recognition," Proc. 7th. DAS, pp. 164-175, Feb. 2006

[6] A. Minagawa, Y. Fujii, H. Takebe, and K. Fujimoto, "Logical Structure Analysis for Form Images with Arbitrary Layout by Belief Propagation," Proc. 9th ICDAR, pp.714-718, Sept. 2007.

[7] E. Oro, and M. Ruffolo, "PDF-TREX: An Approach for Recognizing and Extracting Tables from PDF Documents," Proc. 10th ICDAR, pp.906-910, Jul. 2009.

[8] H. Tanaka, "Threshold Correction of Document Image Binarization for Ruled-line Extraction," Proc. 10th. ICDAR, pp.541-545, Jul. 2009.

[9] S. Shimotsuji, and M. Asano, "Form Identification Based on Cell Structure," Proc. 13th ICPR, pp.793-797.

[10] J. Yuan, L. Xu, and C.-Y. Suen, "Form Items Extraction By Model Matching," Proc. 1st ICDAR, pp. 210-218, Sept. 1991.

[11] T. Watanabe, Q. Luo, and N. Sugie, "Layout Recognition of Multi-Kinds of Table-Form Documents," IEEE Trans. PAMI, vol.18, No.4, pp.432-445, Apr. 1995.

[12] H. Shinjo, E. Hadano, K. Marukawa, Y. Shima, and H. Sako, "A Recursive Analysis for Form Cell Recognition," Proc. 6th ICDAR, pp. 694-698, Sept. 2001.