http://dx.doi.org/10.5392/IJoC.2015.11.4.077

Table Detection from Document Image using Vertical Arrangement of Text Blocks

Dieu Ni Tran, Tuan Anh Tran, Aran Oh, Soo Hyung Kim, In Seop Na*
School of Electronics and Computer Engineering
Chonnam National University, 77 Yongbong-ro, Buk-gu, Gwangju 500-757, Korea

ABSTRACT

Table detection is a challenging problem and plays an important role in document layout analysis. In this paper, we propose an effective method to identify the table region from document images. First, the regions of interest (ROIs) are recognized as the table candidates. In each ROI, we locate text components and extract text blocks. After that, we check all text blocks to determine if they are arranged horizontally or vertically and compare the height of each text block with the average height. If the text blocks satisfy a series of rules, the ROI is regarded as a table. Experiments on the ICDAR 2013 dataset show that the results obtained are very encouraging. This proves the effectiveness and superiority of our proposed method.

***Key words**: Table Detection, Text Block, Expanding ROI, Vertical Arrangement.*

1. INTRODUCTION

Tables, as significant document components, store and present relational information in a condensed way, i.g. experimental results in scientific documents, statistical data in financial reports, price lists, instruction manuals and catalogues etc. Table detection is an important task of document image analysis. Detected table correctly will improve document analysis system and digital library system etc. Table detection and extraction is a popular but difficult problem, primarily due to the diversity of table styles. It is not easy for a single algorithm to perform well on all the difference types of tables.

A wide variety of measures for table detection has been proposed. Jing Fang et al. [1] found that the table headers are one of the main characteristics of complex table styles. They define the lines at the top of a table (header rows) or at the left of the table (header columns) as the table headers. They identify a set of features that can be used to segregated headers from tabular data and build a classifier to detect table headers. In [2], researchers design learning-based framework to identify tables, it is a structured labeling problem, which learns the layout of the document and labels its various entities as table header, table trailer, table cell and non-table region. They develop features which encode the foreground block characteristics and the contextual information. These features are provided to a fixed point model which learns the inter-relationship between the blocks. The fixed point model attains a contraction mapping and provides a unique label to each block.

Yalin Wang et al. [3] define the table detection problem as a probability optimization problem. They proceed to compute a set of probability measurements for each of the table entities. The computation of the probability measurements takes into consideration tables, table text separators and table neighboring text blocks. Then, an iterative updating method is used to optimize the page segmentation probability to obtain the final result. Tanushree Dhiran et al. [4] divide tables into 3 type: table have lines as row and column separator, table have horizontals line for separating rows and space for separating column and tables only space are used as both row and columns separator. They use projection profile and hough line to detected table.

Zhouchen Lin et al. [5] present a robust system which is capable of detecting tables from freestyle online ink notes and extracted their structure so that they could be further edited in multiple ways. First, the primitive structure of tables, i.e., candidates for ruling lines and table bounding boxes, are detected among drawing strokes. Second, the logical structure of tables is determined by normalizing the table skeletons, identifying the skeleton structure, and extracting the cell contents. The detection process is similar to a decision tree so that invalid candidates can be ruled out quickly.

In [6], authors proposed a method to detect table regions in document images by identifying the column and row line separators and their properties. The method employs a run length approach to identify the horizontal and vertical lines present in the input image. From each group of intersecting horizontal and vertical lines, a set of 26 low-level features are extracted and an SVM classifier is used to test if it belongs to a table or not.

Wonkyo Seo et al. [7] develop new junction detection and labeling methods, where junction detection means to find

* *Corresponding author, Email: ypencil@hanmail.net*

Manuscript received Aug. 26, 2015; revised Oct. 19, 2015; accepted Oct. 26, 2015

candidates for the corners of cells, and junction labeling is to infer their connectivity. They consider junctions as the intersections of curves, and so we first develop a multiple curve detection algorithm. After the junction detection, they encode the connectivity information (including false detection) between the junctions into 12 labels and design a cost function reflecting pairwise relationships as well as local observations. The cost function was minimized via the belief propagation algorithm, and they locate tables and their cells from the inferred labels.

Ying Liu et al. [8]-[11] proposed a method in PDF, they noticed that almost all the table rows are sparse lines. By filtering out the non-sparse lines initially, the table boundary detection problem could be simplified into the sparse line analysis problem easily. They design eight line label types and apply two machine learning techniques, Conditional Random Field (CRF) and Support Vector Machines (SVM), on the table boundary detection field. In [12], B. Gatos et al. propose a novel technique for automatic table detection in document images that neither requires any training phase nor uses domain-specific heuristics, thus, resulting to an approach applied to a variety of document types. They propose a workflow for table detection that comprises three distinct steps: image pre-processing; horizontal and vertical line detection and table detection.

Jing Fang et al. [13] propose a novel and effective table detection method via visual separators and geometric content layout information, targeting at PDF documents. The visual separators refer to not only the graphic ruling lines but also the white spaces to handle tables with or without ruling lines and they detect page columns in order to assist table region delimitation in complex layout pages.

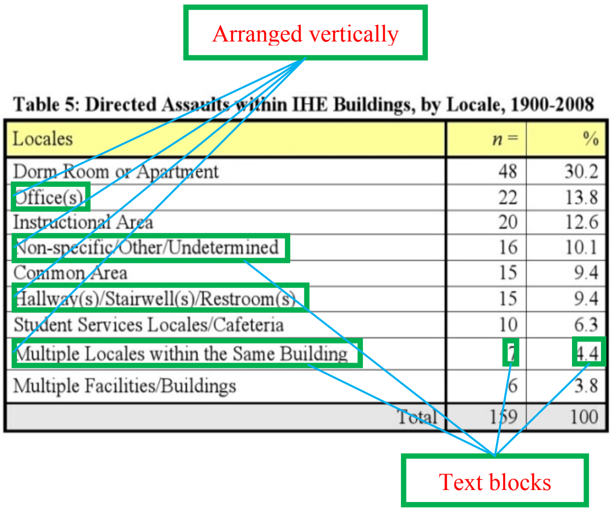
Arranged vertically
Text blocks

Table 5: Directed Assaults within IHE Buildings, by Locale, 1900-2008

Locales	*n* =	%
Dorm Room or Apartment	48	30.2
Office(s)	22	13.8
Instructional Area	20	12.6
Non-specific/Other/Undetermined	16	10.1
Common Area	15	9.4
Hallway(s)/Stairwell(s)/Restroom(s)	15	9.4
Student Services Locales/Cafeteria	10	6.3
Multiple Locales within the Same Building	7	4.4
Multiple Facilities/Buildings	6	3.8
Total	159	100

Fig. 1. Example of the table

Due to the information which can be extracted easily from PDF file, most of proposed method in table detection is proceed on these files. However, text and non-text extraction is not simple in scan document image. To solve this problem, we extract the connected components in the binary image and give some rules to separate text and non-text components.

We observed that the table contains the following properties:

- Contained object: the table is the big object and contains many other objects,
- Arrangement: horizontal lines and vertical lines are usually arranged vertically and horizontally. Text blocks which have the big gaps in same text line and are also arranged vertically, see Fig. 1.

Our method includes the following steps: Firstly, we binarize document image and use morphology to merge neighbor objects. Then, we extract the connected component and get the bounding box of them. After that, we table boundary such as contained object, horizontal line, and vertical line after expand, these are called region of interest (ROI). In ROI, text components are recognized (the height of which is equivalent to the average height). Then, we extract text components in text blocks, which are called table cells. We check text blocks if they are arranged horizontally and vertically. If ROI has many text blocks in a text line and text blocks are arranged horizontally and vertically, then ROI is a real table.

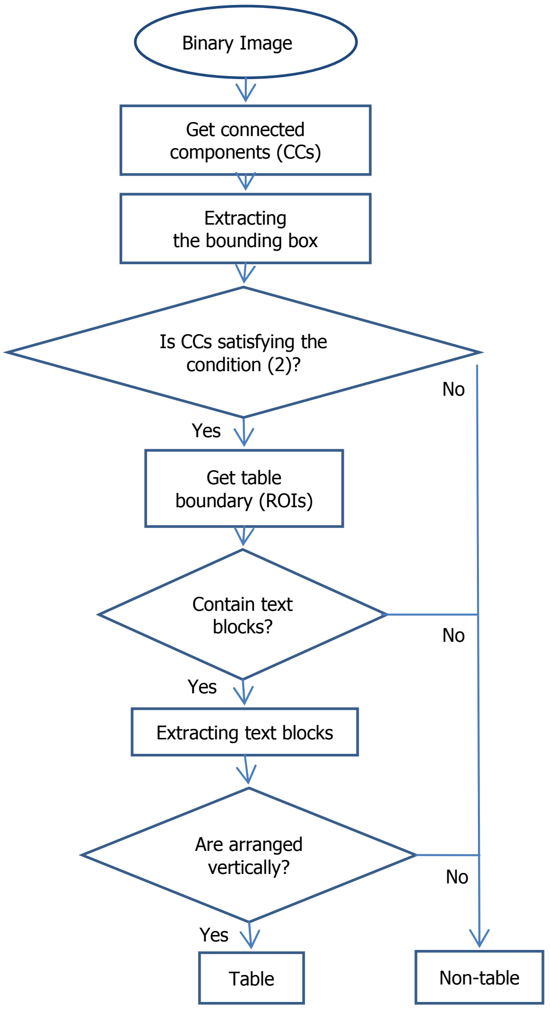
Binary Image
Get connected components (CCs)
Extracting the bounding box
Is CCs satisfying the condition (2)?
No
Yes
Get table boundary (ROIs)
Contain text blocks?
No
Yes
Extracting text blocks
Are arranged vertically?
No
Yes
Table
Non-table

Fig. 2. The flowchart of proposed system

2. PROPOSED METHOD

Like most of image processing method, our method is also implemented on the binary image. Therefore, if the input image is colored, we convert it to the bi-level image by Sauvola algorithm [14]. In this paper, we assign the pixels that belong to the foreground is a value of 1 and background is a value of 0. Then, we apply morphological closing [15] for the given binary image to connect discrete components and reduce noise. Table structure includes the following components: contained objects which are the objects locate inside other objects, horizontal lines and vertical lines, text block, etc. Proposed method uses some equations to locate these properties and gives some conditions to check if they are satisfied tabular attributes. The block diagram of our approach is shown in Fig. 2.

2.1 Extracting connected components and bounding box

Given the binary image, firstly, we extract the connected component CC_s and get their bounding box. Called $CC_i \in CC_s$ is the *ith* connected component and B_i is its bounding box.

The bounding rectangle is detected by x_{left}, x_{right}, y_{top}, y_{bot}. $\forall CC_i \in CC_s$, we consider the number of CC_j which located inside the bounding box of CC_i, see Fig. 5d. The CC_j is called located inside the CC_i if $B_j \subset B_i$, (B_j located inside B_i):

$$Inc(CC_i) = \{B_j \subset B_i | CC_i \in CC_s\} \tag{1}$$

Fig. 3 shows an example of a bounding box. The word "the" is a connected component and it is covered by a bounding rectangle that is located by maxima positions. Fig. 3 also shows *width* and *height* of the connected component.

In this part, we compute the average height of all connected components (*avgHeight*) to use for next steps.

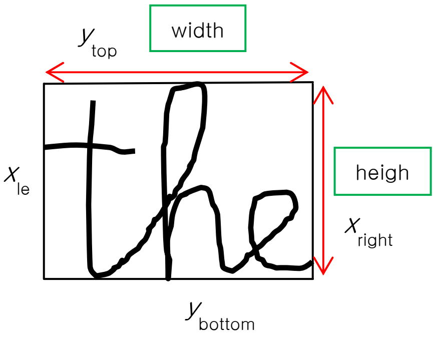

Fig. 3. The bounding box of connected component

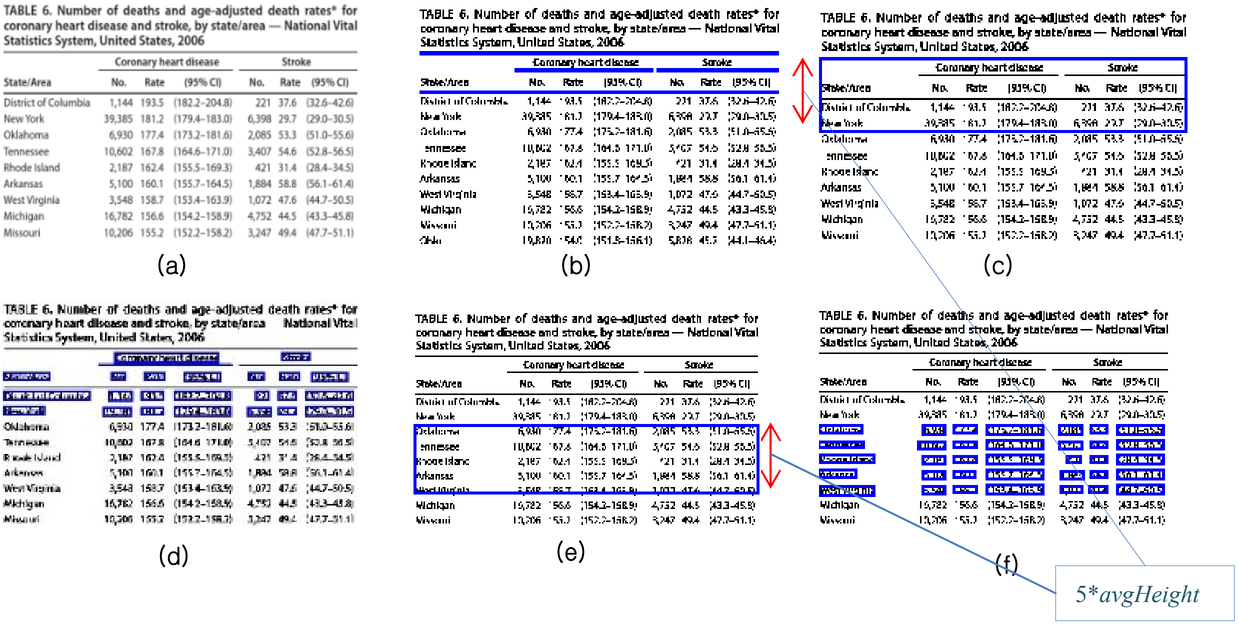
(a)
(b)
(c)
(d)
(e)
(f)
5*avgHeight

TABLE 6. Number of deaths and age-adjusted death rates* for coronary heart disease and stroke, by state/area — National Vital Statistics System, United States, 2006

	Coronary heart disease			Stroke		
State/Area	No.	Rate	(95% CI)	No.	Rate	(95% CI)
District of Columbia	1,144	193.5	(182.2–204.8)	221	37.6	(32.6–42.6)
New York	39,385	181.2	(179.4–183.0)	6,398	29.7	(29.0–30.5)
Oklahoma	6,930	177.4	(173.2–181.6)	2,085	53.3	(51.0–55.6)
Tennessee	10,602	167.8	(164.6–171.0)	3,407	54.6	(52.8–56.5)
Rhode Island	2,187	162.4	(155.5–169.3)	421	31.4	(28.4–34.5)
Arkansas	5,100	160.1	(155.7–164.5)	1,884	58.8	(56.1–61.4)
West Virginia	3,548	158.7	(153.4–163.9)	1,072	47.6	(44.7–50.5)
Michigan	16,782	156.6	(154.2–158.9)	4,752	44.5	(43.3–45.8)
Missouri	10,206	155.2	(152.2–158.2)	3,247	49.4	(47.7–51.1)

Fig. 4. Expanding ROI using our method.
(a) Original document image; (b) The horizontal lines are located as blue lines on binary image; (c) The blue rectangle are located as ROI by expanding the horizontal line and the table height(5**avgHeight*); (d) Extracting text blocks and checking conditions on section 2.3, 2.4 to ROI become a table; (e), (f) Repeating steps (c), (d)

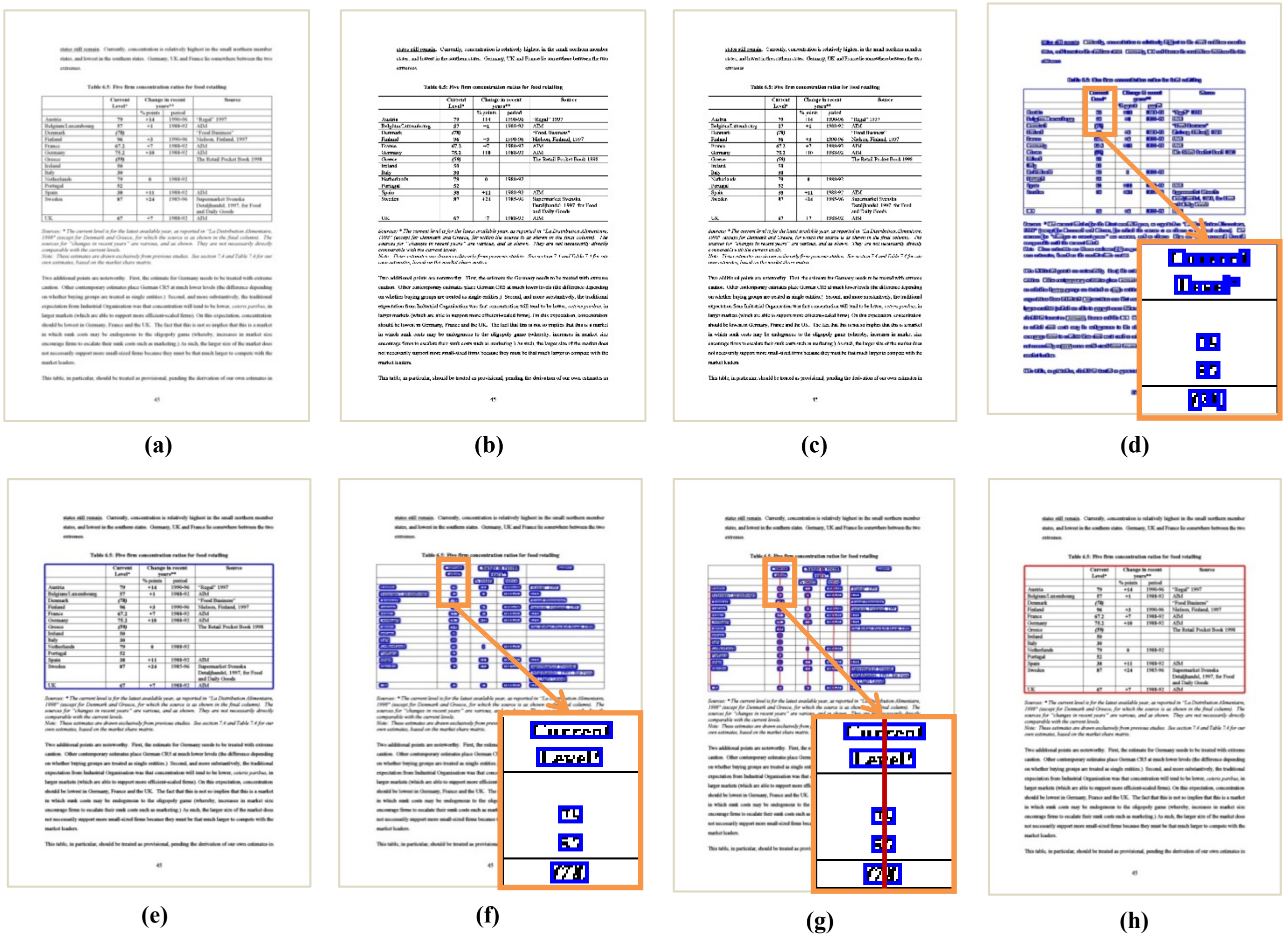
(a) (b) (c) (d) (e) (f) (g) (h)

Figure 5. An illustration of proposed method. We use an orange window to zoom out the result in some steps. (a) Original document image; (b) Binarized document image using Sauvola algorithm; (c) We apply morphological closing for the given binary image to connect discrete components and reduce noise; (d) Extracting bounding box of the connected component in document image; (e) Region of interest (ROI) is detected as contained object; (f) Text blocks is extracting by clustering consecutive text components whose distances are less than a threshold; (g) Verifying ROI is a table by checking vertical arrangement on text blocks; (h) Table is detected.

2.2 Locating region of interest (ROI)

We found that table is an object which contained many rectangle blocks and cells. The boundary of a table could be the contained object or horizontal line, vertical line, etc. Contained object is the big object and its bounding box contains other bounding boxes, see Fig. 5e. This step recognizes contained object from connected component defined as:

$$B_j \subset B_i \Leftrightarrow \begin{cases} x_{left}(CC_i) \leq x_{left}(CC_j) \quad AND \\ x_{right}(CC_i) \geq x_{right}(CC_j)\ AND \\ y_{top}(CC_i) \leq y_{top}(CC_j) \quad AND \\ y_{bot}(CC_i) \geq y_{bot}(CC_j) \end{cases} \qquad (2)$$

where, $B_j \subset B_i$ means CC_j located inside CC_i. Besides, if ROIs are adjacent or overlap, it will be merged together.

There are special cases where the boundary of a table defined by horizontal lines, these tables are called parallel table, Fig. 4a. In these cases, first, we will find these horizontal lines, Fig, 4b, using the height and width of each component. In short, we will use the following condition:

$$width(CC_i) \geq 100 * height(CC_i) \qquad (3)$$

Where $width(CC_i) = x_{right}(CC_i) - x_{left}(CC_i)$,

$height(CC_i) = y_{top}(CC_i) - y_{bot}(CC_i)$

We expand horizontal line from top to bottom using the average height of all connected components. As we know, the table usually has bigger size compared to other objects, hence its height is much greater than the average height many times. From practical experience, we found that table height is greater than the average height at least 5 times.

So, we locate ROI which is detected by the horizontal line as the *width* and expand from top to down with the *height* (5**avgHeight*), see Fig. 4c.

Given ROI, we go to the section 2.3 and 2.4 to verify if ROI is a table, as Fig. 4d. If ROI is not the table, we stop this process. If ROI is the table, we expand again started from the bottom of the latest rectangle (ROI) or the position of latest text blocks, see Fig. 4e-f. These steps are repeated until ROI is not a table.

Therefore, all ROIs, which are detected as the table, are merged together to a united table.

Note that, the expanding process is combined with the locating region of interest process. This means, after every expanding step, the text components inside ROI are checked if they satisfy conditions to become a table. Otherwise, we stop the expanding ROI, see Fig. 4. The detailed condition is given in next step.

2.3 Extracting text blocks

In this step, we determine text components inside ROI. Text elements are components text elements are components whose heights equal to the average height and widths are not too large. We suppose function F guarantees that the two connected components are in the same text line:

$$F(CC_i, CC_j) = \begin{cases} 1 & if\ Y_{top}(CC_i) \leq Y_{bot}(CC_j)\ AND \\ & Y_{bot}(CC_i) \geq Y_{top}(CC_j) \\ 0 & otherwise \end{cases} \quad (4)$$

$\forall\ CC_i,\ CC_j,\ CC_k$; if $F(CC_i, CC_{kj}) = 1$ and $F(CC_j, CC_k) = 1$ then $F(CC_i, CC_k) = 1$; i.e., transitive property holds on relation for connected components.

In each text line, we compute distance of consecutive text components $D(CC_i, CC_j)$, by using the following equation:

$$D(CC_i, CC_j) = \min(\theta_{ij}, \theta_{ji}) \quad (5)$$

where $\theta_{ij} = |x_{left}(CC_i) - x_{right}(CC_j)|$

If the distance of two consecutive text components is less than a gap (threshold), they should be clustered together. We figure out the threshold using below steps:

- Compute the distance of text components in each text line of whole documents.
- Remove values around the *mode* value of the distances set. The *mode* value is the distance between consecutive characters.
- Compute the *mode* value of the rest again and remove values around it. This *mode* value is the distance between consecutive words.
- Compute the variance value of the rest of distances set, γ, this value is the threshold.

Text blocks are text components which are separated by γ. Therefore, each text line has many text blocks and they are also called table cells, see Fig. 5f.

2.4 Checking vertical arrangement

We observed that text blocks (table cells) are always arranged vertically (left side, right side or center), see Fig. 5g. We suppose function *G* guarantees that the two text blocks in others text line are vertical arrangements:

$$\text{G}(CC_i, CC_j) = \begin{cases} 1 & if\ |(X_{left}(CC_i) - X_{left}(CC_j)| \leq \tau\ OR \\ & |(X_{center}(CC_i) - X_{center}(CC_j)| \leq \tau\ OR \\ & |(X_{right}(CC_i) - X_{right}(CC_j)| \leq \tau \\ 0 & otherwise \end{cases} \quad (6)$$

where $X_{center}(CC_i) = \frac{X_{left}(CC_i) + X_{right}(CC_i)}{2}$, τ is deviation.

Note that for any three text blocks $CC_i,\ CC_i, CC_i$; if $G(CC_i, CC_j) = 1$ and $G(CC_j, CC_k) = 1$ then $G(CC_i, CC_k) = 1$; i.e., transitive property holds on relation for connected components. In proposed method, we give value of deviation τ=3.

If ROI has many text lines and text blocks in text lines are arranged vertically, ROI were real table - see Fig. 5h.

The final result contains the location of table boundary and all text cells inside table. All steps of proposed method are shown in Fig. 5.

3. EXPERIMENTS AND RESULTS

In this section, we present the experimental results of our table detection algorithm. The proposed method is able to handle the different table types and give the encourage results.

For the testing, we use the dataset of ICDAR2013 table competition dataset [15] because this dataset is published and very well-known in our field. This dataset contains 77 PDF files table (Fig. 6a), and parallel table which has only horizontal lines (Fig. 6b) or the non-ruling line table, etc.

As mentioned above, our system is implemented on the image instead of PDF file or text file which is given by the competition organizer. Therefore, firstly, we convert all pdf files to images where one page of PDF is one image. Totally, we collect 238 document images (approximate 3 megapixels) with various layouts and different types of table structure.

The evaluation that was proposed by [15] is based on the text regions which located inside the table and the ground truth.

Table 1 shows a result of table detection methods that we refer from ICDAR 2013 Table Competition [15].

Silva et al. [17] give an algorithm that works on textual files line-by-line, and the PDF dataset was therefore converted into text format, resulting in loss of information.

Anssi Nurminen [15] developed the Tabler system that processes born-digital PDF documents using the Poppler library and combines raster image processing techniques with heuristics working on object-based text information obtained from Poppler in a series of processing steps.

Burcu Yildiz developed the pdf2table system [18] which employs several heuristics to recognize tables in PDF files having a single column layout. For multi-column documents, the user can specify the number of columns in the document via a user interface; however, such user input was not allowed in the competition. The approach was able to handle most of the documents where the tables span the entire width of the page.

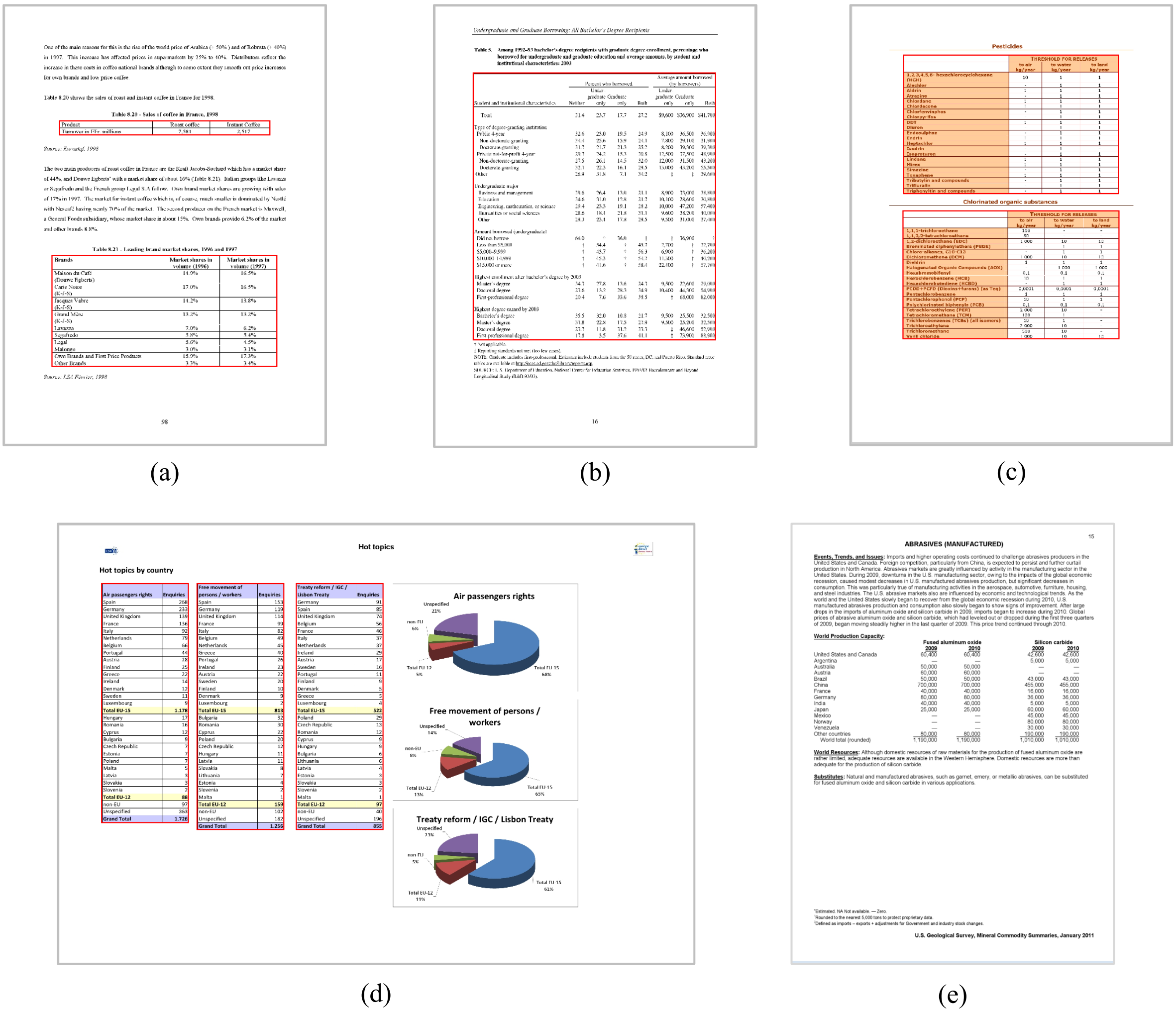

Fig. 6. Examples of proposed method, the table detection are marked by red rectangle.
(a) Fig. 6a shows a normal case of table detection using contained object to locate the ROI; (b) There is parallel table, which of table are boundary by only horizontal lines. We process expanding the horizontal line and identify the tabular characteristics until cannot find them anymore. Finally, all ROIs, which are detected as the table, are merged together to a united table; (c) With color table, the images after binary are lost a lot of information about text block. However, we provide a robust method to determine what is table for the rest; (d) Fig. 6d shows some table near each other but the distances not too close and our system detect correctly; (e) Fig. 6e shows a failure case of our method. This table have no boundary information, hence we cannot locate ROI due verify this region is a table.

Table 1. Result for table detection

Author	**Type**	**Per-document averages**			**Tables found (total=156)**	
		Recall	***Precision***	***F_1-measure***	***Complete***	***Pure***
Proposed	***Image***	***0.9636***	***0.9521***	***0.9578***	***147***	***141***
Silva [17]	PDF	0.9831	0.9292	0.9554	149	137
Nitro [15]	PDF	0.9323	0.9397	0.9360	124	144
Nurminen [15]	PDF	0.9077	0.9210	0.9143	114	151
Acrobat [15]	PDF	0.8738	0.9365	0.9040	110	141
Yildiz [18]	PDF	0.8530	0.6399	0.7313	100	94
Stoffel [19]-[20]	PDF	0.6991	0.7536	0.7253	79	66
Liu et al [11]	PDF	0.4601	0.3666	0.4080	39	95
Hsu et al [15]	Image	0.2697	0.7496	0.3967	28	41

Andreas Stoffel et al. [19]-[20] participated with a trainable system for the analysis of PDF documents based on the PDFBox library. After initial column and reading-order detection, logical classification is performed on the line level. In order to detect tables, the system was trained on the practice dataset using a sequence of a decision-tree classifier and a conditional random field (CRF) classifier. Consecutive lines labelled as tabular content were then grouped together and output as a table.

William H. Hsu et al. [15] proposed The Kansas Yielding Template Heuristic Extractor (KYTHE) which is designed to process scanned documents by using an OCR tool such as Tesseract. The approach combines automatic preprocessing (using lists of expected attributes and template-based constraints) with interactive post-processing, enabling the system to be adapted for a specific data source.

The TableSeer system [11] was developed by Ying Liu. The algorithm uses a heuristic approach by first joining together adjacent text lines with uniform font size, before using whitespace and textual cues to determine which blocks contain a table.

There are many commercial systems join this competition such as Adobe Acrobat XI Pro, Nitro Pro 8, etc. Acrobat system loads each document and saved as HTML. The region result file was manually generated based on the content of the result tables. Max Gobel et al. [15] use The "To Excel" conversion function of Nitro outputs all detected tables in Excel format (one file per document; one worksheet per page). The given results are very encouraging.

As shown at Table 1, for image documents, we get higher detection rates compared to the other methods. The F1-measure for all text cells is 95.78%, while the Recall is 96.36% and Precision is 95.21%. Our system detects 147 complete tables and 141 pure tables. The correctly table is 140 tables which are both complete tables and pure tables.

$$F_1 - measure = 2.\frac{Precision.Recall}{Precision + Recall} \quad (7)$$

In table a region is classified as complete [16] if it includes all sub-objects in the ground truth region; a region is classified as pure if it does not include any sub-objects which are not also in the ground truth region. A correctly detected region is, therefore, both complete and pure.

Some results of table detection using our method are shown in Fig. 6. Fig. 6a shows a normal case of table detection using contained object to locate the ROI. In case of Fig. 6b, there is a parallel table, which of table is boundary by only horizontal lines. We process expanding the horizontal line and identify the tabular characteristics until cannot find them anymore. Finally, all ROIs, which are detected as the table, are merged together to a united table. With color tables are shown in Fig. 6c, the images after binary are lost a lot of information about text block. However, we provide a robust filter to determine what a table for the rest is. Fig. 6d shows some table near each other but the distances not too close and our system detect correctly. Fig. 6e shows a failure case of our method. The table on this image does not have any boundary information, so our system cannot locate ROI.

We also test on the dataset of 44 images that are scanned from printed paper. The proposed method detects 62/67 tables. The global performance metric for all images is 92.53%. In this dataset, some of images are blurry, so binary images have noises and lose most of the information about boundaries.

4. CONCLUSIONS

In this paper, we have proposed an algorithm for detecting tables from scanned documents. Our system has the advantage of better time consuming and results on the ICDAR 2013's dataset.

Proposed method works on scanned document image instead of PDF file as some previous approaches, so it is more challenging. Our method focuses on ROI instead of the whole document due to it have better time consuming. Our system also handles in case of parallel tables which have only horizontal lines as table boundaries. Our method also has a good result on images with multi-column. We proposed a new approach that checks vertical arrangement of text blocks to verify a table.

Proposed method base on a boundary of a table so it has a bad result detect in cases tables without boundary or complex backgrounds such as a color table, table overlap with text or image.

In the future, we extend method in cases of a table which has no boundary information. In addition, we handle cases touching among text component and table boundary. With the color table, we will binary the color region using multi-threshold to get the clear result.

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2015-018993) and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2015R1C1A1A02036495).

REFERENCES

[1] Jing Fang, Prasenjit Mitra, Zhi Tang, and C. Lee Giles, "Table Header Detection and Classification," Association for the Advancement of Artificial Intelligence, 2012, pp. 599-605.

[2] Anukriti Bansal, Gaurav Harit, and Sumantra Dutta Roy, "Table Extraction from Document Imag es using Fixed Point Model," Indian Conference on Computer Vision Graphics and Image Processing, 2014, Article no. 67.

[3] Yalin Wang, Ihsin T. Phillips, and Robert M. Haralick, "Table Detection via Probability Optimization," Document Analysis Systems V, pp. 272-282.

[4] Tanushree Dhiran and Rakesh Sharma, "Table Detection and Extraction from Image Document," International

Journal of Computer & Organization Trends, vol. 3, issue 7, Aug. 2013, pp. 275-278.

[5] Zhouchen Lin, Junfeng He, Zhicheng Zhong, and Rongrong Wang, "Table detection in online ink notes," IEEE Trans Pattern Anal Mach Intell, 2006, pp. 1341-1346.

[6] T Kasar, P Barlas, S Adam , C Chatelain, and T Paquet, "Learning to Detect Tables in Scanned Document Images Using Line Information," Document Analysis and Recognition (ICDAR), 2013, pp. 1185-1189.

[7] Wonkyo Seo, Hyung Il Koo, and Nam Ik Cho, "Junction-based table detection in camera-captured document images," International Journal on Document Analysis and Recognition 2015, pp. 47-57.

[8] Ying Liu, "A Fast Preprocessing Method for Table Boundary Detection: Narrowing Down the Sparse Lines using Solely Coordinate Information," Document Analysis Systems, DAS '08, The Eighth IAPR International Workshop on, 2008, pp. 431-438.

[9] Liu Ying, Mitra Prasenjit, and Giles C. Lee, "Identifying table boundaries in digital documents via sparse line detection," 17th ACM conference on Information and knowledge management, pp. 1311-1320.

[10] Ying Liu, Kun Bai, Prasenjit Mitra, and C. Lee Giles, "Improving the Table Boundary Detection in PDFs by Fixing the Sequence Error of the Sparse Lines," Document Analysis and Recognition, 2009- ICDAR '09, pp. 1006-1010.

[11] Ying Liu, Kun Bai, Prasenjit Mitra, and C. Lee Giles, "TableSeer: automatic table metadata extraction and searching in digital libraries," JCDL '07 Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries, pp. 91-100.

[12] B. Gatos, D. Danatsas, I. Pratikakis, and S. J. Perantonis, "Automatic Table Detection in Document Images, Pattern Recognition and Data Mining," Lecture Notes in Computer Science, vol. 3686, 2005, pp. 609- 618.

[13] Jing Fang, Liangcai Gao, Kun Bai, Ruiheng Qiu, Xin Tao, and Zhi Tang, "A Table Detection Method for Multipage PDF Documents via Visual Seperators and Tabular Structures," 2011 International Conference on Document Analysis and Recognition, pp. 799-783.

[14] J. Sauvola and M. PietikaKinen, "Adaptive document image binarization," Pattern Recognition 33, 2000, pp. 225-236.

[15] Max Gobel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi, "ICDAR 2013 Table Competition," 2013 12th International Conference on Document Analysis and Recognition, pp. 1449-1453.

[16] Rafael C. Gonzalez, Richard E. Woods, and Prentice Hall, *Digital Image Processing (3rd Edition)*, 3 edition (August 31, 2007), Chapter 9 Morphological Image Processing, pp. 627-680.

[17] A. C. e Silva, *Parts that add up to a whole: a framework for the analysis of tables*, Ph.D. dissertation, The University of Edinburgh, 2010.

[18] B. Yildiz, K. Kaiser, and S. Miksch, "pdf2table: A method to extract table information from pdf files," in IICAI, 2005, pp. 1773-1785.

[19] H. Strobelt, D. Oelke, C. Rohrdantz, A. Stoffel, D. A. Keim, and O. Deussen, "Document cards: A top trumps visualization for documents," IEEE Trans. Vis. Comput. Graph, vol. 15, no. 6, 2009, pp. 1145-1152.

[20] A. Stoffel, D. Spretke, H. Kinnemann, and D. A. Keim, "Enhancing document structure analysis using visual analytics," in SAC, 2010, pp. 8-12.

Dieu Ni Tran

She received the B.S in Mathematics & Computer Science from Ho Chi Minh City University of Science, Vietnam in 2013. Her main research interests include pattern recognition, image processing, text recognition, document segmentation.

Tuan Anh Tran

He received the BS degree in Mathematics and Computer Science, University of Science, Ho Chi Minh city, Viet Nam, in 2010 and the MS degree in Apply Mathematic in MAPMO, University of Orleans, France, in 2011. He is currently researching as a Ph.D student at Electronics and Computer Engineering, Chonnam National University, Korea. His research interests include document layout analysis, pattern recognition, machine learning, and mathematics application.

A Ran Oh

She received her B.S. degree in school of Computer Statistics from Chosun University, Korea in 2009, She is currently researcher at Department of Computer Science Chonnam National University, Korea.

Soo Hyung Kim

He received his B.S. degree in Computer Engineering from Seoul National University in 1986, and his M.S. and Ph.D degrees in Computer Science from Korea Advanced Institute of Science and Technology in 1988 and 1993, respectively. From 1990 to 1996, he was a senior member of research staff in Multimedia Research Center of Samsung Electronics Co., Korea. Since 1997, he has been a professor in the Department of Computer Science, Chonnam National University, Korea. His research interests are pattern recognition, document image processing, medical image processing, and ubiquitous computing.

In Seop Na
He received his B.S., M.S. and Ph.D. degree in Computer Science from Chonnam National University, Korea in 1997, 1999 and 2008, respectively. Since 2012, he has been a research professor in Department of Computer Science, Chonnam National University, Korea. His research interests are image processing, pattern recognition, character recognition and digital library.