논문 2013-1-7

AN EFFICIENT METHOD FOR DOCUMENT IMAGE GEOMETRIC LAYOUT ANALYSIS

Yunkoo Chung*

Abstract

Document image analysis is necessary for optical character recognition (OCR) and also very useful for many other document image manipulations. In this paper, we propose a document image geometric layout analysis system which has less region segmentation and classification error than that of the commercial software and previous works. The proposed method segments the document image into small regions to the size of a character using fast connected components generation method, so that it prevents the different types of connected components from combining. We also propose new criterion for clustering the connected components and some new techniques to deal with noise and reduce computation time. Experiment shows classification error rate of text and picture regions is decreased.

Key Words : Document Image Analysis, Connected Component Analysis , Optical Character Recognition(OCR)

1. Introduction

Analysis and interpretation of document images is important for OCR and also very useful for many other document image manipulation purposes such as document storage, transmission, retrieval and understanding. The system which automatically converts paper document into electronic document, consists of three parts which geometric and logical document layout analysis, character recognition and document format converter. Of these, geometric document layout analysis which segments and classifies the document image into detailed regions such as text, picture, table, bar and so on, has a lot of effect on the performance of this system. Because the result of this part is used in the rest of the system. If the region classification error occurred in this part, paper document would be converted into different shape of electronic one. However, it is not easy to reduce the classification error of the text and the picture regions because the size, the density and pixel distribution of a picture region are so diverse that these features of some picture regions are similar to those of the text. Methods for document image segmentation can be divided into two main categories: bottom-up or data-driven approaches[1-5], and top-down approaches[6-8]. Top-down approaches rely on a document model and thus usually are restricted to certain types of documents, such as that blocks must be enclosed in rectangles or uniform column arrangement, etc. Texture is also applied[9] although it requires a lot of computation time. In this paper, we propose a document image geometric layout analysis system which has less

* ETRI(Electronics and Telecommunications Research Institute
(email: ykchung@etri.re.kr)
접수일자: 2013.6.7 수정완료: 2013.6.26

region segmentation and classification error than that of the commercial software and previous works. The proposed method segments the document image into small regions to the size of a character using fast connected components generation method, so that it prevents the different types of connected components from combining. We also propose new criterion for clustering the connected components and some new techniques to deal with noise and reduce computation time. This method has little influence on the size and the pixel distribution of a region, so a small text region can be classified even though the region consists of one character. And this method also has misclassified text regions into pictures but reverse cases are rare, so misclassified regions can be corrected by the classification error correction routine. Therefore, classification error rate of text and picture regions are decreased.

2. Connected Component Analysis

There are a large number of connected components in a document. Before component analysis, we reduce the image to about 50dpi. This reduces the component number and thus reduces the time complexity of the system.

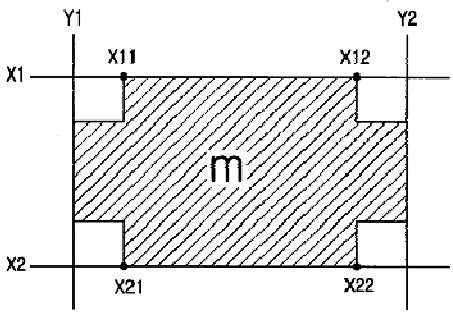


Fig. 1. Connected component

We then scan the reduced image row by row, and merge any 8-connected runs in successive rows. A set of merged runs is a connected component. A connected component is represented by *(y1, y2, x1, x2, x11, x12, x21, x22, m),* where *y1, y2, x1, x2* are the horizontal and vertical extents of the bounding rectangle; *x11*, *x12* are the leftmost and rightmost *x* coordinate in the top row respectively, *x21*, *x22* are the leftmost and rightmost *x* coordinate in the bottom row respectively and *m* is the area (number of black pixels)
The following is the algorithm for connected component analysis:

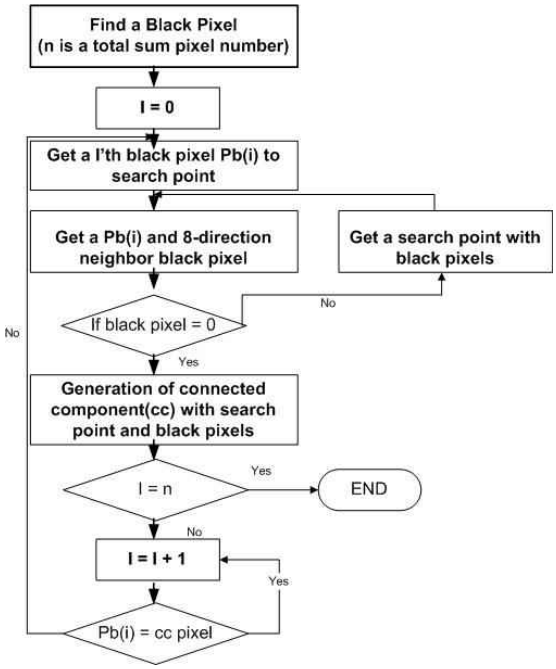


Fig. 2. Generation of connected components using 8 directional search

In noisy documents, it is possible that characters in adjacent rows (horizontally aligned text) or adjacent columns (vertically aligned text) are connected in the reduced image. Since our method is bottom-up, it is necessary to split these connected components. For every component, the original image in the bounding rectangle of the connected component is projected to Y-axis and X-axis. If at some point the projection is small, the connected component is split. From the vertical and horizontal projection, we also refine the position of bounding rectangle.

3. Rough Classification of Components

It is suitable to represent a document structure as a tree. The whole document page is the root. Each internal node represents a meaningful block such as a table, a text block or a photo, etc.. Each leaf is a connected component. Fig. 8(b) shows

segmented blocks the document image in Fig. 8(a). Fig. 9(c) shows the tree description of the page in Fig. 9(a).

We construct an initial tree from the connected components. For connected components *i* and *j*, *j* is *i*'s father if *j* is sufficiently large and the overlap between the bounding rectangles of *i* and *j* is larger than half of the bounding rectangle of *i*. Connected components in a table, frame or photo is grouped into a single node with its embedded text, while connected components in the text block bounded by white space will be clustered in the next step.

Then, nodes are roughly classified. A connected component with large height and small width is a vertical bar. A connected component with large height and large area (black pixel number) is considered as vertical photo. Similarly, we classify horizontal bar and horizontal photograph. A connected component whose width and height are over the largest characters is also non-text, which may be table, frame or photos (the method for table analysis is presented elsewhere). Other components are considered as possible text.

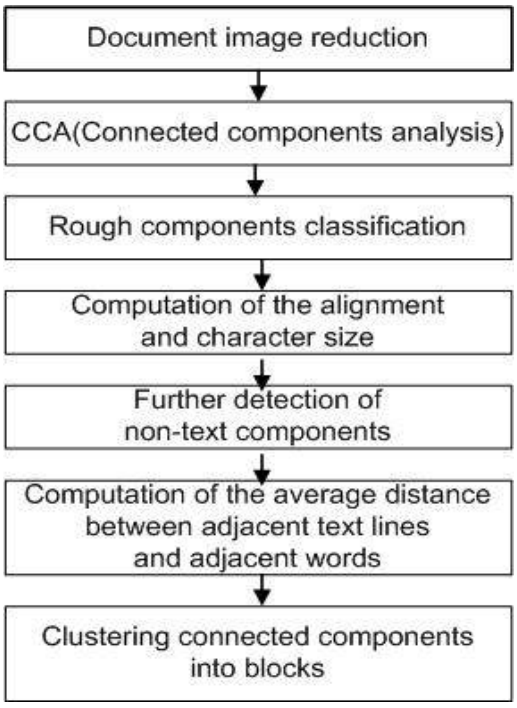


Fig. 3. Block diagram of document image geometric layout analysis method

4. Clustering of Text Components

We have generated an initial tree for the document. In the tree, components in some block such as a table, a large photo or bar is grouped with its text into a single node, while other text components are direct sons of the root. Components in a text block separated by white spaces are not grouped. In this section, we cluster the text components according to the white space between them. The clustering is performed on possible text components that have the same father node. To deal with document mixed with horizontal and vertical aligned text and non-uniform spaced text,, the computation of the alignment, character size and space between text in adjacent text rows or adjacent words is performed for each internal node instead of the whole document.

4.1 Computation of the Alignment and Character Size

In the reduced document image, characters may be connected. In horizontally (vertically) aligned text, the space between adjacent rows (columns) is usually larger than space between adjacent characters in the same row (column). Thus adjacent characters in the same row (column) are more likely connected, while characters in different rows (column) are separated. We therefore can determine the orientation of an internal node according to the statistics on the height and width of all its connected components.

For every internal node, we average the height and width of all its connected components whose width or height is in possible range of character size. A connected component whose both height and width are in the range is counted twice. Let this average be *smeans*, which approximates the character size. Let *HSizeN* and *WsizeN* be the number of the components whose height and width approximates to *smeans* respectively. If *HSizeN* > *WsizeN,* the node is horizontal-aligned, otherwise the node is vertical- aligned.

After the block orientation is determined, we estimate the character size of characters, which will provide an important measure for succeeding operations. If the document is horizontal-aligned, we calculate the height of the characters as the size, otherwise we calculate the width as size.

For simplicity, from now on, we only discuss the method for horizontal-aligned documents. Vertical-lined documents can be processed symmetrically.
For the calculation of the height of characters, a histogram of the height of connect components in the node whose heights are in the possible range and approximate to *smeans* is constructed with *2mm* as intervals. We then locate the peak and calculate the average height of the components whose height is in the peak, with the width of the component as weight. This average weight is considered as the average character size of the node.

4.2 Computation of the Average Distance between Adjacent Text Lines and Adjacent Words

With the character size in each internal node, we further detect some non-text connected components. The process is similar in section 3. The only difference is here we use the character size as a measure. Before we cluster the connected components, we calculate the average distance between adjacent text rows.
Since documents often skew, the distance between two adjacent rows is not equal to the difference between the Y-positions of two character lines. Moreover, to obtain the text line is just our aim. We estimate the distance between two lines from every component. For a connect component, we search for other components that overlap with it in X-direction. The smallest Y-distance from it to those components is considered as the estimation of its distance to adjacent lines. A large value of the estimation may not correspond to space between adjacent rows and thus is discarded. Fig. 4 shows an example.

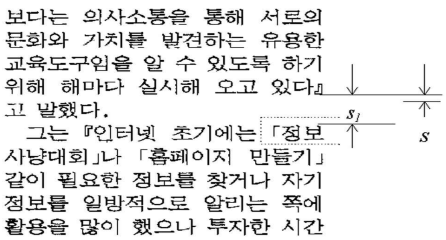


Fig.4. Average distance between adjacent lines.

We then average these estimation for all text components. This average is considered the distance between adjacent lines. Since a connected component is usually much smaller than a text line, we can obtain the distance more precisely from individual connected components than directly from text lines. Similarly, we calculate the distance between two adjacent words.

4.3 Detection of Connected Component Type

Fig. 5shows Single connectivity. If max(rc.left, rp.left) <= min(rc.right, rp.right) then rp and rc are 8-connected extend the blob

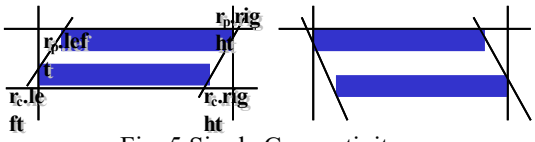


Fig. 5 Single Connectivity

Fig. 6 shows find a two adjacents line and merge to y-th line of connected component and y+1-th line of connected component with share position parents and children. The algorithm continues to merge the whole image and generation of completely form of connected components.

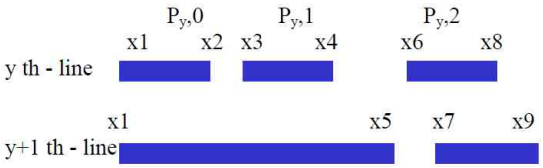


Fig. 6 Example of the connected components between neighbored two lines

In the merge process to parents and children, we use the following equations for decision to merge with two adjacent lines.

$$\text{Min}[ex(P_{y,m}),ex(P_{y+1,n})] \geq \text{Max}[sx(P_{y,m}),sx(P_{y+1,n})] \quad \cdots\cdots\cdots (1)$$

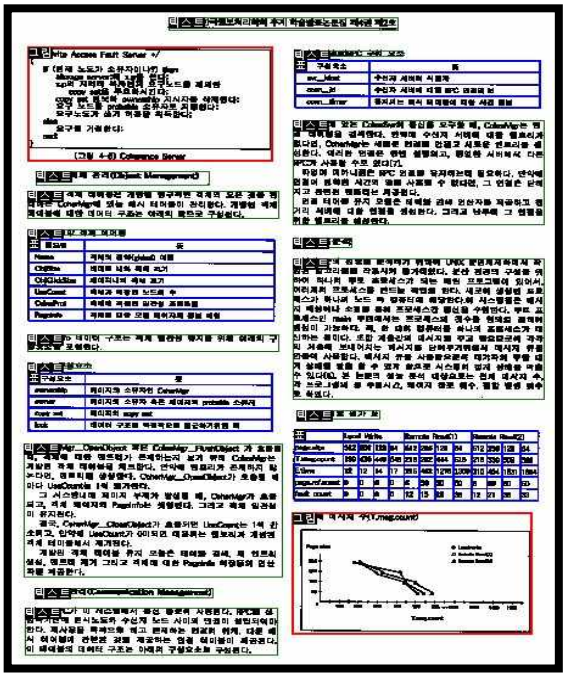
Where max[a,b] is maximum value of a,b and min[a,b] is minimum value of a,b. sx[c] is start coordinate of connected component C and ex[c] is end coordinate of connected component C

4.4 Clustering Connected Components into Blocks

For the possible text components that have the same father node, we cluster them into text blocks. The clustering is based on the distance between two components. Any components that are sufficiently close are clustered into one block. Knowledge-based rules are used to determine if two components are sufficiently close. Generally, if the vertical distance of the bounding rectangles of two connected components is small compared to the distance between adjacent lines and character height, and those two rectangles overlap in X-direction, we say they are close. Similarly, if the horizontal distance of the bounding rectangles of two connected components is small compared to the distance between adjacent words and character width, and those two rectangles overlap in Y-direction, we also say they are close. Then we group a connected component into a block if it is close to any components of the block. If a component is not close to any components, it is assigned to a new block. The newly grouped block is inserted into the tree as an internal node between its components and the father of its components.

5. Refinement of Text Blocks by Iterative Split and Merge

In Section 4, we cluster connected components into blocks according to the alignment, character size and space between adjacent line which are estimated on a whole node. However, a node may initial contains several blocks. In different blocks these parameters may be different. It may result in errors in the clustering. We therefore introduce a refinement. As a block is formed, we calculate the text alignment, the space between character rows or column and the character size in it. The calculation is the same as in Section 4. These parameters are more precise estimation than in their father node. With these more precise estimation, we further exclude more non-text components(a text block with too small size is discarded), check if a block should be split and if two blocks should be merged. For every block, if the distance between some two adjacent rows in the block is sufficiently large compared to the average distance between adjacent rows, we partition it into two. If two blocks that have the same father are close enough, we group it into one. If any component is excluded as non-text, a block is split, or two blocks are merged, the process is performed again on each node, forming an iterative process. Since we deal with each block individually, the method is applicable to mixed aligned text.



6. Experiment and Conclusion

We have applied this algorithm to six page images, with results shown in Table 1. For this set of images ,selected from a variety of journals with various styles, some with mixed format (vertically and horizontally aligned). The good performance owes to the high accuracy of the approximation of character size and distance between adjacent rows (columns) and adjacent words and the iterative refinement.

This algorithm contribute promising results to most of newspaper, journals, textbook images. By using those schemes together, most document images accurately. Future work will consider more difficult conditions such as high noise and more kind of fonts.

Type of classification region	character	picture	table	line	sum
Num of Real region classification	404	121	32	29	586
Result of classification	396	111	29	26	562
Ratio of classification	98%	92%	91%	90%	96%

Table 1. The result of region classification

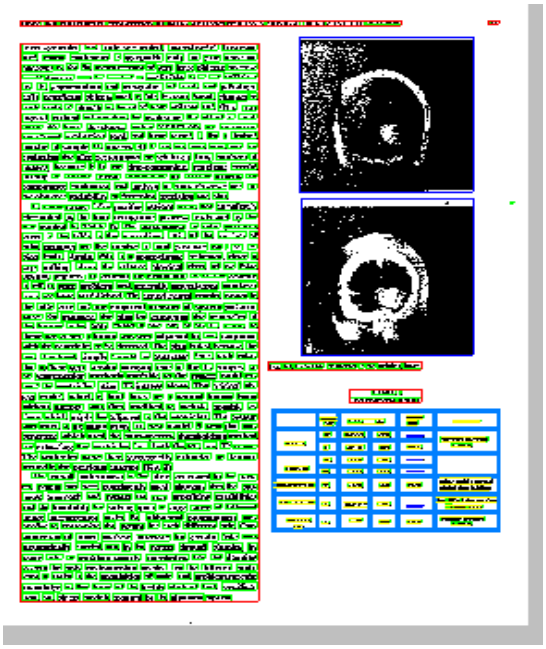


Fig.7. Example of region classification

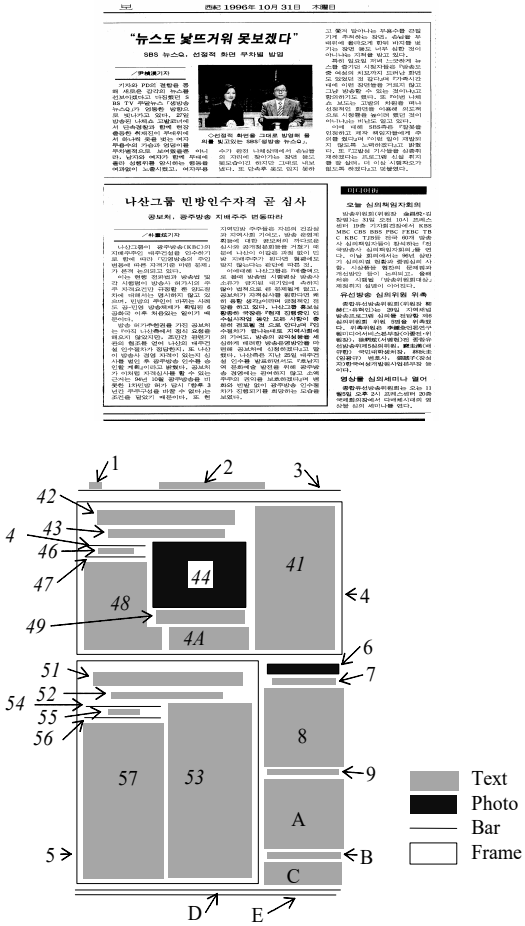
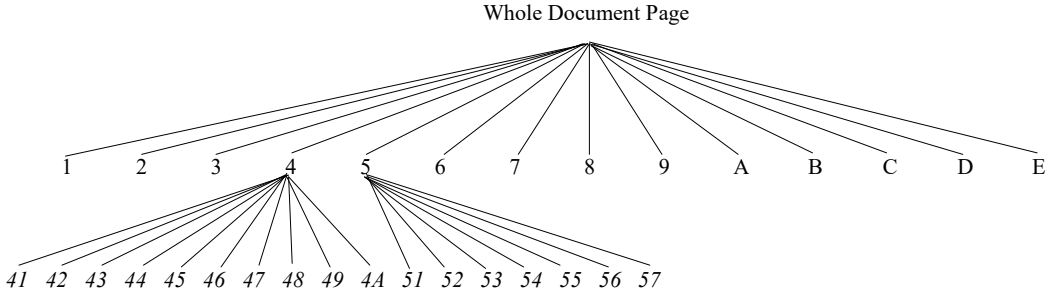


Fig.8. A Document Image and Segmented Result



(c) The tree description

Fig. 9. A document image and its tree description

References

[1] Sin-Ywan Wang and Toshiaki Yagasaki, Block segmentation: a method for segmenting page image of various editing styles, *Int. Conf. on Document Analysis and Recognition*, Montreal, Canada, 1995, 128-133.

[2] Hones F and Lichter J., Layout extraction of mixed mode documents, *Machine Vision and Application, Vol. 7*, 1994, 237-246.

[3] L. A. Fletcher and R.K. Kasturi, A roubust algorithm for text string separation from mixed text/graphics images, *IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol.10, No.6*, Nov. 1988, 910-918.

[4] T. Pavlidis and J. Zhou, Page segmentation by white stream, *Proc. 1st Int. Conf. On Document Analysis and Recognition, Saint-Malo, France, Sept.*1991,945-953.

[5] K. Kubota, O. Iwaki and H. Arakawa, Document Understanding System, *Proc. 7th Int. Conf. On Pattern Recognition*, 1984, 612-614.

[6] G. Nagy et al, A prototype document image analysis system for Technical Journals, *IEEE Computer, Vol. 25, No. 7*, 1992, 10-22.

[7] Masaharu Ozaki, Column segmentation by white space pattern matching, *Int. Conf. on Document Analysis and Recognition, Montreal*, Canada, 1995, 128-133.

[8] D. Drivas and A. Amin, Page Segmentation and Classification Utilizing Bottom-up Approach, *Proc. ICDAR,* 1995, 610-614

[9] Wang D. and Srihari S. N. Classification of newspaper image blocks using texture analysis, *CVGIP, Vol. 47*, 327-352.

저 자 소 개

Yunkoo Chung

YUN KOO CHUNG obtained his undergraduate degree in Electronics Engineering from Korea University (Korea) in 1979, his M.D. in Computer Science from Cleveland State University (USA) in 1985, and his Ph.D. in Computer Science from Wayne State University (USA) in 1991. Currently he is a principal researcher in ETRI since 1991. His research areas are pattern recognition for document character and 3D object shape and image processing for embedded digital cameras of PDAs.