8th IPPR Conference on
Computer Vision, Graphics and Image Processing
Tao-Yuan, Taiwan, R.O.C.
Aug. 20~23, 1995

Segmentation of Texts, Graphics, and Special Components for Color Document Image Analysis

Yen-Huei Chuang (莊豔輝) and Dr. Wen-Hsiang Tsai (蔡文祥)
Institute of Computer and Information Science
Department of Computer and Information Science
National Chiao Tung University, Hsinchu, Taiwan, Republic of China

Abstract

A system for segmentation of texts, graphics, and special components in color document images is proposed. First, a modified center-cut color quantization method is proposed, which can be used to classify color image pixels into several classes. Then methods for combining subimages into complete images, adjusting skewed document images, and detecting print orientations of document images are proposed. For block segmentation, a multi-spectral constrained run length algorithm is used to extract objects from multiple color planes. After block extraction, several statistical features and a structural feature are used to classify text blocks and graphic blocks. Then methods for merging text blocks into text areas are proposed. Finally, methods for detecting the print orientations of text areas and for classifying their types are proposed. Some experimental results are shown to prove the feasibility and practicability of the proposed approach.

1. Introduction

1.1 Survey of Related Studies

Optical scanning is a common method for capturing data from a paper document. The scanned data are stored in a file of picture elements, calls pixels. These pixels have values 0 or 1 for binary images, 0 to 255 for gray-scale images, and three channels of 0 to 255 color values for color documents. But a document image contains only raw data; it must be analyzed further to gather desired information.

Techniques for document analysis have been studied for many years. In the mid to late 1980s, document image analysis began to grow rapidly. Commercial document analysis systems are now available for storing business forms, performing optical character recognition on typewritten texts, compressing engineering drawings, etc. Document analysis research continues to pursue more intelligent handling of documents, better compression, and faster processing.

A general document processing usually includes four basic steps. The first step is image preprocessing, which includes binarization, noise reduction, skewed image adjustment, etc.

The second step is block segmentation. The objective is to partition an image into distinct regions of homogeneous data. Several methods for segmenting gray-scale document images have been proposed in the literature. But few methods are proposed for color document images. The proposed methods for gray-scale images can be broadly classified as either top-down or bottom-up approaches. Most top-down techniques are based on the run length smoothing (RLS) algorithm (Wahl et al. [1]), and the projection profile cut algorithm (Wang and Srihari [2]). And bottom-up methods (Fletcher and Kasturi [3]) typically involve grouping of pixels as connected components which are merged into successively larger regions. Except for the above two types of approaches, Jain and Bhattacharjee [4] proposed a multichannel filtering-based texture segmentation method which can be used for text-graphics separation. Two-dimensional Gabor filters are used to compute texture features. Recently, the neural network is used for segmentation of document images as well. For color images, Lin and Tsai [5] proposed a multi-spectral constrained run length algorithm (MCRLA) which can segment blocks in parallel in all the color planes. This method is modified from the run length smoothing algorithm.

After block segmentation, the next step is to recognize the block type. If a block is recognized as text lines, it need be merged further into a larger block. The final step is document understanding. The objective is to extract the logical relationships between the texts, graphics, and other components. In this study, we investigate mainly the second and the third steps.

1.2 Overview of Proposed Approach

In real cases, many color document images are quite complicated. In order to reduce the complexity of the processing work, some assumptions are made in this study, as described in the following.

1. We do not process overlapping graphics.
2. A graphic is assumed to be of rectangular shape ; graphics with irregular shapes are not processed.
3. Only newspapers and magazines are processed.

When a color document is read in, the first stage of our approach is color quantization. Color is important information in images. However, it will put heavy burden on image processing if a true color image is processed. The purpose of color quantization is to reduce the number of colors and classify the colors. A modified center-cut color quantization algorithm is proposed. After color quantization, the next stage is document image creation,

including adjustment of image orientations and merge of multiple subimages. The largest size that the scanner can scan is fixed. Therefore, when a large document is scanned, it must be separated into several subdocuments. To keep the information of an entire document, merge of multiple subimages is needed. At first, the system checks if the image need be combined. If so, each of the subimages is binarized. Then the system detects the skew angle of each subimage. When any of the subimages is skewed, the system adjusts its orientation. After finishing the adjustments of all subimage orientations, the system merges the subimages into a complete image. If no image merge is required, the system checks the skew angle of the input image directly and adjusts it. When detecting the skew angle, we can also detect the orientations of most printed texts. If the orientations of the texts are vertically, we call this kind of document as vertical document. If the orientation of the texts are horizontally, we call it as horizontal document. In the following steps, we adopt different methods for different types of documents. As for object extraction, to reduce storage space and processing time, we adopt the multi-spectral constrained run length algorithm (MCRLA) proposed by Lin and Tsai [5]. For vertical documents, we process it in the vertical direction. For horizontal documents, we process it in the horizontal direction. At first, the system extracts special components, including vertical lines, horizontal lines, frames, and tables. Then, the objects of all colors are extracted concurrently. After the objects are extracted, the object types are recognized. We use some features to classify the objects into graphic blocks and text blocks.

After color quantization, the characters of an identical color often cannot be perfectly classified into one class. Parts of one character might also be classified into several classes. However, parts of a text block separated into different colors often overlap with one another. We should merge such overlapping text block parts into one text block. After the above process, neighboring textline blocks are merged further into larger blocks.

After all the blocks in a document are extracted, we remove the noise that may occur on the text or graphic blocks. Some text and graphic blocks are difficult to separate. So we need some ways to remove the noise coming from erroneous classifications. Finally, we combine larger blocks into text areas and detect the orientations and types of the text areas.

2. Color Quantization

2.1 Survey of Related Studies

A method of color quantization is usually designed to recursively partition the color space into quantization cubes. Each of the cubes is assigned a representative color. And all the colors within a cube are regarded to form a class and reproduced by the representative color value. The method of median-cut color quantization is often used. And the method of mean-cut color quantization and center-cut color quantization are modified from median-cut color quantization. The main principle of the three methods are similar. We review them in the following.

Initially, the entire color space is regarded as a single cube whose representative color is the mean vector of the r, g, and b components. The histogram for each color component is then created and the variances of the histograms are computed by

$$Var(R) = \sum_{r=\inf}^{\sup} p(r) \cdot |r - mean\_r|;$$
$$Var(G) = \sum_{g=\inf}^{\sup} p(g) \cdot |g - mean\_g|; \quad (1)$$
$$Var(B) = \sum_{b=\inf}^{\sup} p(b) \cdot |b - mean\_b|$$

The distribution with the largest variance is split into two parts. The cut points are the median, mean, or center of the cube for median-cut, mean-cut, or center-cut color quantization, respectively. After the partitioning, a cube is divided into two smaller cubes, and all the color coordinates with similar values in each cube are clustered together. An appropriate threshold value T is preselected to measure the quality of the quantized image. If the variance of the cube is smaller than threshold T, then it is not split any more. Otherwise, splitting is performed recursively.

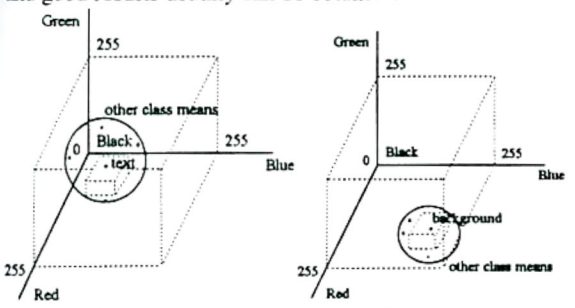
2.2 Proposed Modified Center-Cut Color Quantization

The main color of the texts in a document is usually black, and the color of the background is usually white. Analyzing the color histogram distributions, we see that the main color of the texts and the color of the background are always separated on the two sides of the histogram. The method of center-cut color quantization [6] can classify color pixels more correctly. But it is still not good enough. So we proposed a method for use in this study that is modified from the center-cut color quantization method.

The method consists of two steps. The first step is to perform conventional center-cut color quantization, and the second step is to merge quantized classes whose means are close enough into one class. The first step have been introduced previously. The second step is described in the following.

After conventional center-cut color quantization is performed, we compute the pixel count and the mean for each class. The class that has the maximum pixel count is taken to be the class of the background color. And the class that has the second largest pixel count is taken to be the class of the main text color. Furthermore we compute the distance between the class of the main text and each of the other classes. If the distance is small enough, we merge the two classes into one (see Fig. 1(a)). Also, to remove some noise, we compute the distance between the class of background and each of the other classes. If the distance is small enough, we merge them (see Fig. 1(b)). Then we compute the distances between each pair of the other classes. If the distance is small enough, we also merge

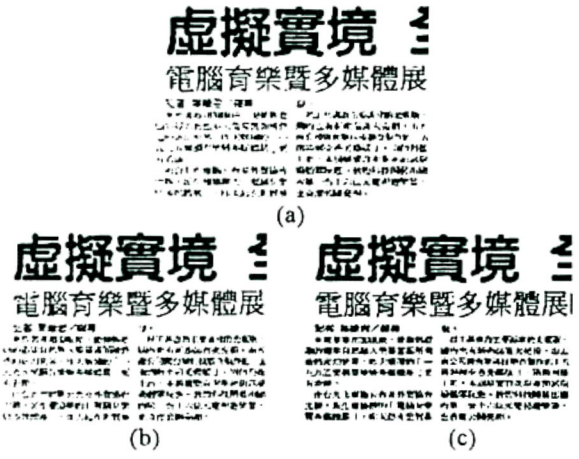
them. After the above two steps, all pixels are classified and good results usually can be obtained.



(a) (b)

Fig. 1. Merge of main text class and background class with other classes in modified center-cut color quantization. (a) Merge of the class of the main text and other classes. (b) Merge of the class of background and other classes.

As an example of experimental results, the pixels in the class of the main text after performing mean-cut, center-cut, and modified center-cut color quantization on a piece of a newspaper article image are shown in Fig. 2(a), Fig. 2(b), and Fig. 2(c), respectively. From these figures, we can observe obviously that the method of modified center-cut color quantization is better than the method of mean-cut color quantization and the method of center-cut color quantization.



(a) (b) (c)

Fig. 2. The pixels in the class of the main text. (a) After performing mean-cut color quantization. (b) After performing center-cut color quantization. (c) After performing modified center-cut color quantization.

3. Document Image Creation

3.1 Image Orientation Adjustment by Hough Transform

3.1.1 Detection of skew angle and orientation of document

We use the Hough transform [7] to detect the skew angle of a document which is assumed not to exceed $\pm 10^o$ So for vertical documents, the range for detecting the skew angle is from 80^o to 100^o. And for horizontal documents, the range is from -10^o to 10^o. But we initially do not know the orientation of the document. We must detect all the possible angles in the two ranges. For each black pixel in row x, column y of the image, we perform the following steps :

$$\begin{aligned} &\text{for } \theta = -10^\circ \text{ to } 10^\circ \text{ and } \theta = 80^\circ \text{ to } 100^\circ \\ &\quad \text{compute } \rho = x\cos\theta + y\sin\theta, \text{ and} \\ &\quad \text{set } accumulator[\rho,\theta] = accumulator[\rho,\theta] + 1. \end{aligned} \tag{2}$$

Thus, after all the pixels in the given document image are considered, votes in the accumulators correspond to the strength of evidence for a straight line to exist in the document with the corresponding parameters. Note that a point in the document image corresponds to a sinusoidal curve in the parameter space, and a point in the parameter space corresponds to a straight line in the document image.

If a document is skewed along a particular orientation θ_o, when the ρ values in the accumulator array are scanned for each orientation θ, it can be found that the maximum number of low-high-low transitions of the ρ values will occur for the orientation θ_o. A high value of ρ corresponds to a textual line and a low value corresponds to a white space in between the textual lines. For a document which is not skewed, the situation should occur at 90^o and 0^o. But for vertical documents the situation occurring at 0^o will be more obvious than occurring at 90^o, as shown in Fig. 3. And for horizontal documents this situation occurring at 90^o will be more obvious than occurring at 0^o. From this phenomenon, we can determine whether the document is vertical or horizontal. The result obtained by performing the Hough transform on an image is the same as computing the projection profile of the image (Srihari and Govindaraju [8]). Thus, after computing the Hough accumulator array values, if we sample the column values at each θ_i, i=1, 2, . . . , k, in the accumulator array, the values will contain the same result as obtained from projecting the profile of the image onto a line that is making an angle θ_i with the horizontal axis. When θ_i is very close to the true orientation of the document image, the textual lines cause regular and repaid fluctuations in the projection profile, as shown in Fig. 3(c). For orientations that are not close to the true orientation, the fluctuations are more gradual (Baird 1987 [9]), as shown in Fig. 3(d). Therefore, we can detect the skew angle by computing the sum of squares of the accumulator values for each angle θ_i as follows:

$$R(\theta_i) = \sum g(\rho_j, \theta_i)^2, \qquad j = 1, \ldots, \rho\_size \tag{3}$$

where $g(\rho_j, \theta_i)$ is the value at accumulator array location (ρ_j, θ_i).

After computing all $R(\theta)$, we can find the rough skew angle θ. Then we search the range from θ-1 to

$\theta+1$ with the increment of 0.1^o for a more precise skew angle, again using the Hough transform.

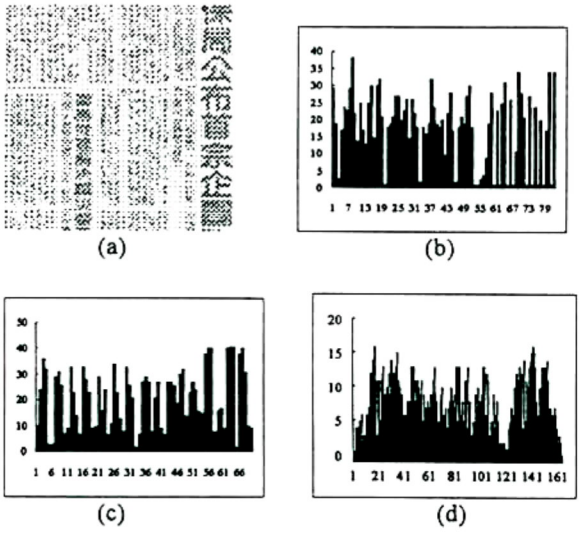


Fig. 3. Skew angle detection. (a) Source image. (b) All possible values of ρ in Hough accumulator array shown as a histogram when $\theta = 90^o$. (c) All possible values of ρ when $\theta = 0^o$. (d) All possible values of ρ when $\theta = 5^o$.

The algorithm that we adopt to find the skew angle and the orientation of a document is as follows:

Stage 1: Detection of skew angle.

Step 1: Detection of rough skew angle.

Let θ_r = maximum $R(\theta)$

where $-10^o \le \theta \le 10^o$ *or* $80^o \le \theta \le 100^o$.

Set rough skew angle = θ_r.

Step 2 : Detection of exact skew angle.

Let θ_e = maximum $R(\theta)$

where $\theta_r - 1 \le \theta \le \theta_r + 1$.

Set exact skew angle = θ_e.

Step 3: Determine the skew angle.

If exact skew angle $\ge 80^o$,

Set skew angle= $\theta - 90^o$;

else

Set skew angle= θ.

Stage 2: Detection the orientation of document.

If skew angle $< 80^o$, then

set the document orientation to be vertical;

else

set the document orientation to be horizontal.

3.1.2 Image orientation adjustment

After detecting the skew angle, if the document is skewed, we use back transformation to adjust the image orientation. Using back transformation can avoid discontinuous results in the new image. For each pixel in row x', column y' of the new created image, we compute

$$x = x' \cos\theta - y' \sin\theta$$
$$y = x' \sin\theta + y' \cos\theta \quad (4)$$

where θ is the detected skew angle and the values x and y are the row and column positions in the old image. Then we assign the r, g, and b values of pixel (x, y) in the old image to that of pixel (x', y') in the new image. If the pixel (x, y) does not exist in the old image, we assign the color of background to pixel (x', y').

3.2 Merge of Multiple Subimages

When we process a document that cannot be scanned once, we must split the document into several subdocuments. Each subdocument overlaps with its adjacent subdocuments. We then scan each subdocument separately. After scanning, we must merge the subimges into a single larger one .

The method that we use is template matching. There are two cases to deal with. One is that we have two subimages to merge. The other is that we have four subimages to merge. In case one, the system first asks the user about the orientation of merge, namely, vertical or horizontal. If the orientation of merge is horizontal, we extract the rightmost block of subimage 1 and the leftmost block of subimage 2. The width and the height of an extracted block are 30 and a half of the image height, respectively. If the orientation of merge is vertical, we extract the bottom block of subimage 1 and the top one of subimage 2.

Then, one of the two blocks is repeatedly moved in the horizontal and vertical direction in order to find the best position for merge according to computed correlation coefficient values. Then, we merge the two subimages into one image. In the above process, if the orientation of merge is horizontal, we call the process as "HC" (horizontal combination). Otherwise, we call it as "VC" (vertical combination).

In case two, we have to merge four subimages-subimage 1, subimage 2, subimage 3, and subimage 4. At first, we perform the "HC" process that has been mentioned in case one to merge subimage 1 and subimage 2. Then, we perform the "HC" process to merge subimage 3 and subimage 4. Finally, we perform the "VC" process to merge subimage 1 and subimage 3. There are three combination positions, so we can merge the four subimages into a complete one.

4. Segmentation of Basic Blocks and Special components

In our study, we use the multi-spectral constrained runlength algorithm (MCRLA) proposed by Lin and Tsai [5] to extract basic blocks and special components. Its main idea is the same as the runlength smoothing algorithm [1]. The difference is that the MCRLA is proposed for color document image.

4.1 Extraction of Special Components

Initially, we use the MCRLA to find horizontal runs in the horizontal direction. If the length of a horizontal run is larger than a threshold value, we keep it. We discard runs which are too short. After collecting runs, each run is checked to see whether it is adjacent to any of the corresponding block embryos with the same color spectra. If it is, we merge them. When the entire image are processed, we can get horizontal lines. If a horizontal line is too thick, we remove it because such a line could be part of a graphic or a text block.

As to the extraction of vertical lines, the method is the same as the extraction of horizontal lines. The only difference is that we perform the MCRLA in the vertical direction.

After extraction of vertical and horizontal lines, the next step is to extract the frames and tables. For each vertical line and horizontal line, we find the junction of them. If the junction is close the top end of the vertical line and the right end of the horizontal line, it means that the two lines meet at a corner of a frame. We merge the two lines into a frame. If the junction is in the middle of the vertical line or in the middle of the horizontal line, it means that the two lines are part of the lines of a table. We merge the two lines into one table.

After all tables and frames are extracted, the lines that are not part of the frames and tables are regarded as independent lines. The lengths of them should not be too short. If the length of a line is smaller than a threshold value, we remove the line; it may be part of the text or part of a graphic.

4.2 Extraction of Basic Blocks

For vertical documents, we perform the MCRLA to extract the basic blocks in the vertical direction at first. And for horizontal documents, we extract the basic blocks in the horizontal direction. After extraction of basic blocks, the following processes are the same for both types of documents.

5. Basic Block Recognition

5.1 Features of Basic Blocks for Recognition

The features we use for recognition can be separated in two types. One is the type of statistical features. The other is the type of structural feature proposed by Lin and Tsai [5]. These features are described in the following.

5.1.1 Statistical features

(1) The saturation degree of a block :

$$C = \frac{number\ of\ black\ pixels}{area\ of\ surrounding\ rectangle}. \qquad (5)$$

(2) The compactivity degree :

$$C = \frac{number\ of\ isolated\ pixels}{number\ of\ black\ pixles}. \qquad (6)$$

For each black pixel, we compute the number of the neighbors with a 3x3 mask. If the number is zero, the pixel is regard as an isolated pixel. Then we compute the value of C by (6).

(3) The connectivity degree :

$$N = \frac{number\ of\ connected\ pixels}{number\ of\ black\ pixels}. \qquad (7)$$

For each black pixel, we compute the number of the neighbors with a 3x3 mask. If the number is larger than 5, the pixel is regard as a connected pixel. Then we compute the value of N by (7).

5.1.2 Review of Structural Feature Proposed by Lin and Tsai [5]

The main idea of this feature is based on the fact that all words are produced by lines whose widths are identical. This feature is good for smaller fonts of words. But for larger fonts, the characters are constituted by runs with different line widths. So the average line width may be far from those of runs. This feature will be called the linewid feature. For details, see [5].

5.2 Recognition of Basic Text and Graphic Blocks

The rules we use for recognition of basic text and graphic blocks are as follows.

Rule 1:
IF the threshold values of the statistical feature value are not satisfied,
THEN the type of the block is GRAPHIC.

Rule 2:
IF the value of the linewid feature is high enough and the value of the connectivity degree is not too low,
THEN the type of the basic block is TEXT.

Rule 3:
IF the value of the connectivity degree is high enough and the value of the linewid feature is not too low,
THEN the type of the basic block is TEXT.

Rule 4
IF all the above rules are not satisfied,
THEN the type of the basic block is GRAPHIC.

6. Text Block Construction

6.1 Introduction

The texts in documents that people see are black. However after image scanning, around the edges of characters always show other colors. Sometimes the color of a character edge will be changed from black to gray gradually. These situations always occur in small-font texts. We call these colors that are not real colors in the document as false colors. Therefore, no matter what methods of color quantization are used, the text will be

classified into more than one color class. After performing the MCRLA, for a single character there is one basic block for each class. We call those basic blocks that are produced by false colors as false blocks.

6.2 Merge of False Blocks

To handle a false block, the type of the block must be recognized at first. If the type of the false block is text and the false block overlaps with other basic text blocks, we merge these blocks into one textline block. Also, a textline block may touch other textline blocks. If a textline block overlaps with other textline blocks, we merge them into one textline block.

If the type of a false block is graphic and it overlaps with other basic graphic blocks, we merge these basic graphic blocks into a larger graphic block. Also, a graphic block may overlap with other textline blocks. This kind of graphic block is not really graphic and is regarded as noise. Such noise will be removed later.

6.3 Merge of Textline Blocks into Larger Blocks

The method of merging textline blocks includes two phases, merge of textline blocks in the vertical direction, and merge of textline blocks in the horizontal direction, and is described as follows:

Phase 1: Merge of textline blocks in the vertical direction.

Because the size of a punctuation mark is smaller, textline blocks are always broken by the positions of punctuation marks. Therefore, for a Chinese document one textline block is always separated into several textline blocks at the positions of punctuation marks. For convenience of future works, we must merge these broken textline blocks at first.

For each textline block, we check whether any textline block overlaps with it in the vertical direction. If so, two cases need be treated. One case is that the line widths of the two textline blocks are very large. For this, we check if the distance between the two textline blocks is smaller than a threshold value and if the orientation of the two blocks are both vertical. If so, we merge them. The distance threshold value is set to the maximum character width of the two textline blocks. The other case is that the line widths of the two textline blocks are not very large. For this, we check whether the two textline blocks satisfy the following conditions.

1. The colors of the two blocks are the same.
2. The difference of the line widths computed from the structural features of the two blocks is small.
3. The difference of the average character widths in the two blocks is small.
4. The distance between the two textline blocks is smaller than a threshold value. If the above four conditions are satisfied, we merge the two textline blocks into one.

Phase 2: Merge of the textline blocks in the horizontal direction.

If the textline block does not satisfy the conditions of Phase 1, Phase 2 is performed.

The goal of Phase 2 is to merge texline blocks into larger blocks. For each texline block, we check if it overlaps with some larger blocks in the horizontal direction. If so, two cases need be treated. One case is that the line widths of the two textline blocks are very large. For this, we check if the distance between the two textline blocks is smaller than a threshold value and if the orientation of the two blocks are both horizontal. If so, we merge them. The distance threshold value is set to the maximum character width of the two textline blocks. The other case is that the line widths of the two textline blocks are not very large. For this, we check if the two blocks satisfy several conditions. These conditions are the same as the four conditions in Phase 1. If the above conditions are satisfied, we further check if they should be merged. There are four possible cases, as shown in Fig. 4.

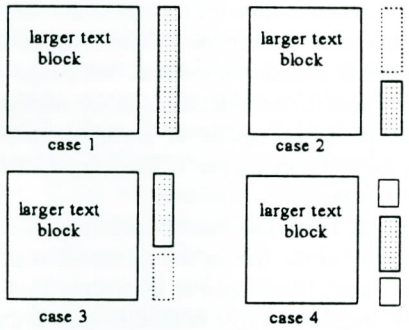


Fig. 4. Merge of textline blocks into larger blocks. There are four cases that might occur. The block of dashed lines may be nonexistent .

In case one, we merge the textline block and the larger text block into an even larger text block. In case four, we do not merge the textline block and the larger text block into a larger text block. If case two or case three occurs, we set a flag to the larger text block. We first check the flag of the larger text block. If the flag is set on, we merge the textline block and the larger text block into a larger text block and reset the flag off. Otherwise, we do not merge the two blocks and set the flag on. The reason for setting flags is that the height of the textline blocks on the same paragraph might not be the same. We must separate them into two larger text blocks.

6.4 Noise Removal

The noise found in our study includes two types. The first type comes from the case that some text blocks are recognized as graphic blocks. The other type comes from the case that some graphic blocks are recognized as text blocks. The first type occurs because of false colors in the edges of text. And the second type occurs because of erroneous recognition of text and graphic blocks. A method of removing these types of noise is described in the following.

6.4.1 Removing noise of graphic blocks

Two characteristics of graphic blocks are: (1) the size of a graphic block are large; (2) the ratio of the length over the width is not too large.

Using these characteristics, we can remove false graphic blocks. At first, for each graphic block, we check the size of it. If the size of the graphic block is small, we discard it. If it is large enough, we check the ratio of its length over its width. If the ratio is larger than a threshold value, we remove it. If the two conditions are not satisfied, we check further if the graphic block overlaps with other text blocks. If so, we check the sizes of the graphic block and the text block. If the size of the graphic block is not much larger than the size of the text block, it means that the graphic block is noise, which is then discarded.

6.4.2 Removing noise of text blocks

For each text block, we check the size of it. If the size is too small, we discard it. If the size is large enough, we check if the text block is in the graphic block. If it is, we increase the threshold values of the features to check the type of the block again. If each feature value of the text block is larger than a new threshold value, the type of the block is still text. Otherwise, we merge the text block with the graphic block.

6.5 Recognition of Text-Area Types

For each text area, we must recognize the type of it. The types include the HEAD of an article and the TEXT of an article. For the type of HEAD, a characteristic is that it has a larger line width or a larger average character width. So for any text area satisfying the above characteristic, the type of the text area is decided to be HEAD. Otherwise, the type of the text area is TEXT.

6.6 Detection of Text-Area Orientation

The method we use for detecting the text-area orientation can be separated into two cases. One is for the text area of the HEAD type. The other is for the text area of the TEXT type. For the HEAD type, we compute the ratio of the height over the width. If the ratio is larger than 2, the orientation of the text area is decided to be vertical. Otherwise, the orientation of the text area is horizontal. But for the text area of the TEXT type, we perform a vertical projection and a horizontal projection. If the text area is vertically printed, the interval between two textline blocks in the vertical projection will be larger than the interval in the horizontal projection (see Fig. 5). Using the size of the interval, we can decide the orientation of the text area.

7. Experimental Results

Several images obtained from newspapers and magazines were tested in our system. The system was implemented on a Sun Sparc 10 workstation based on the C language. And the tested images were obtained by an Umax vista s-8 color scanner at 250 dpi. The segmentation results are shown in Fig. 6 and Fig. 7. The results for extracting basic blocks and extracting special components can be seen to be good.

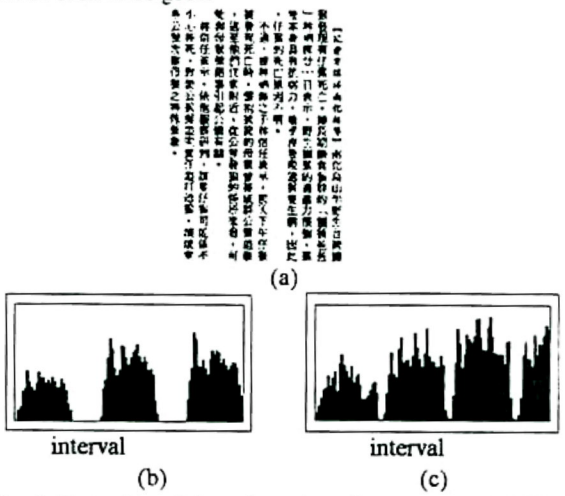


Fig. 5. Detection of the orientation of a text area. (a) The text area (b) The vertical projection. (c) The horizontal projection.

8. Conclusions

An image segmentation system for color documents has been successfully implemented. Several achievements in different processing stages are summarized as follows.

In the stage of color quantization, a modified center-cut color quantization method has been proposed. We can classify color pixels more effectively. In the stage of document image creation, we have proposed methods for adjusting the orientation of a skewed image, detecting the orientation of printed text, and merging subimages into complete images.

Then for each color plane, basic blocks and special components are extracted by the MCRLA. And text blocks and graphic blocks are classified using a recognition method which combines the usages of statistical features and a structural feature.

In the stage of text block construction, methods for merging false blocks and merging basic text blocks into larger text blocks have been proposed. Also, methods for removing noise in text blocks and graphic blocks were introduced. And the methods for recognizing the types of text areas and detecting the orientations of text areas have been proposed.

To sum up, it is advantageous to extract the objects by color information. Good experimental results have proved the feasibility of our approaches.

References

[1] F. M. Wahl, K. Y. Wong, and R. G. Casey, "Block Segmentation and Text Extraction in Mixed Text/Image Documents," *Computer Graphics and Image Processing 20*, 375-390, 1982.

[2] D. Wang and S. N. Srihari, "Classification of newspaper image blocks using texture analysis," *Computer Vision, Graph. and Image Process.*, Vol. 47, 1989, pp.327-352.

[3] L. A. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 10, 1988, pp. 910-918.

[4] Jain, A. K. and Bhattacharjee S., "Text segmentation using Gabor filters for automatic document processing," *Machine Vision and Applications*, Vol. 5, 1992, pp. 169-184.

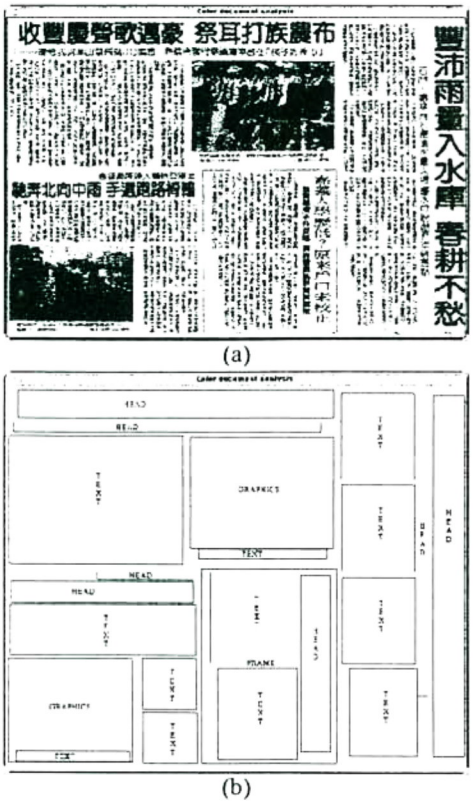
[5] Y. S. Lin and W. H. Tsai, " Image Segmentation for color Document Analysis," in *Proc.. Conf. Computer Vision, Graphics and Image Processing*, NanTou, Taiwan, R.O.C., 1994, pp. 135-142.

[6] G. Joy and Z. Xiang, "Center-cut for color-image quantization," *Visual Computer*, Vol. 10, 1993, pp.62-68.

[7] Hough P. V. C, "Method and means for recognizing complex patterns," *US Patent* 3,069,654, December 18, 1962.

[8] S. N. Srihari and V. Govindaraju, "Analysis of Textual Images Using the Hough Transform," *Machine Vision and Applications*, Vol. 2, 1989, pp.141-153.

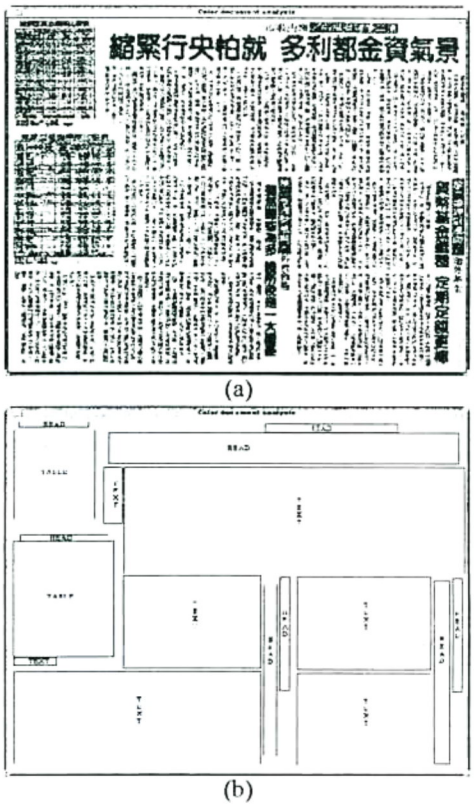
[9] H. S. Baird, "The skew angle of printed documents", *Proceedings of Society of Photographics Scientists and Engineers*, Vol. 40, 1987, pp21-24.



(a)

(b)

Fig. 6. Segmentation result I. (a) The source color image with size 2950 x 2456. (b) The result of segmentation and block recognition.



(a)

(b)

Fig. 7. Segmentation result II. (a) The source color image with size 2684 x 2474. (b) The result of segmentation and block recognition.