Automatic Electronic Book Construction from Scanned Page Images by Image Analysis Techniques

Jung-Kuang Hsu(許榮光) and Wen-Hsiang Tsai(蔡文祥)

Department of Computer & Information Science
National Chiao Tung University
1001 Ta Hsueh Rd., Hsinchu, Taiwan 300, R.O.C.
Tel: 886-3-5712121 Ext. 56650
Email: whtsai@cis.nctu.edu.tw

Abstract

Based on the use of image analysis techniques, a system for automatic construction and display of electronic book contents from scanned page images is proposed. First, the Hough transform is employed to estimate the skew angles of scanned page images, and adjust their orientations. The resulting image is then decomposed into basic blocks using moment-preserving thresholding and region growing techniques. An automatic method is proposed next to estimate the threshold parameters for the size and point density features of the basic blocks. These parameters are used to classify basic blocks into three types: text, graphic, and line blocks. Furthermore, a systematic repetitive pattern detection method is proposed to find common parts among different page images, which improves book content compression. For page images with pictorial backgrounds, also proposed is a method for extracting texts on pictorial backgrounds to gain more text information. Finally, a user-friendly interface is designed for displaying the electronic book content according to a proposed book content organization structure.

Keywords: electronic book, basic block, pictorial background, repetitive pattern, moment-preserving thresholding.

1 Introduction

1.1 Motivation

Electronic books are becoming more and more popular for reading various kinds of digital contents. An advantage of the use of electronic books is its ease for archiving and retrieval, compared with printed books. Therefore, automatic digitization of printed book contents into electronic forms for use in electronic book applications has become an important research subject.

Usually, the pages of a book contain pictures and line drawings, along with texts. An important procedure in digital image processing of books is page layout analysis whose goal is to segment texts, graphics, and line drawings out of page backgrounds. Appropriate parameter values involved in the analysis yield better page image segmentation results. The parameters usually are selected after observing intermediate processing results in most studies [1-3]. If wrong parameter values are selected, erroneous page image analysis results will be produced. But up to now, there still lack systematic and automatic methods for calculating these parameters, especially for complex page contents or pictorial backgrounds, so it is desired in this study to find a solution to such a problem.

Data compression is essential for efficient transmission and archiving. After the layout of a given page image is analyzed, different types of components in the image will be obtained. Many algorithms have been proposed to compress these components respectively in single page images [3, 4]. But few methods have been proposed to compress all the page images of an entire book. In a book, there often exist various types of common parts among different page contents in a book. Saving these common parts only once will improve further the content compression effect. It is desired to detect such repetitive patterns in books systematically, and save them efficiently in this study.

In summary, in this paper we try to design a system to digitize and display book contents by image analysis techniques, and propose algorithms to calculate image-processing parameter values automatically, to handle repetitive patterns systematically, to segment page components effectively, etc.

1.2 Overview of Proposed System

First we use an automatic document feeder (ADF) to scan all pages in a book into images. Image orientation adjustment is next conducted on the scanned page images. Moment-preserving thresholding and region growing [5, 6] are then applied to find all the basic blocks of the page images. Two types of features are used to classify the basic blocks into three types, namely, text, graphic, and line drawing. We also propose methods to calculate automatically the threshold parameters for using these features in basic block classification.

After all blocks in the page image are

classified, different methods are applied to process them. For text blocks, we propose a method to merge text blocks into text frames without overlapping the graphic blocks. Each text frame contains multiple text lines, and we segment each text line to obtain the characters in the line. Then we apply an OCR system to recognize the characters in the text frame [9]. For graphic blocks, the JPEG compression algorithm [7, 8] is applied to compress them. For line blocks, their positions and thickness are extracted and stored. When an electronic book is displayed, a graphic line is drawn at the original position to replace the original image line. In addition, there exist common parts, called repetitive patterns in this paper, between the pages in the book. We propose a hierarchical method to detect repetitive patterns and store them to improve the compression rate further.

When a page image includes a pictorial background, there often exist texts on the background to describe the content of the picture. We also propose a method to extract texts on pictorial backgrounds, and restore the pictures.

After the above processes are performed, the page images of the book are converted into the electronic format. We also set up a book data structure and provide further a friendly interface for displaying and reading the electronic book pages, based on the use of the book data structure. The overall procedure of the system operations is shown in Fig. 1.

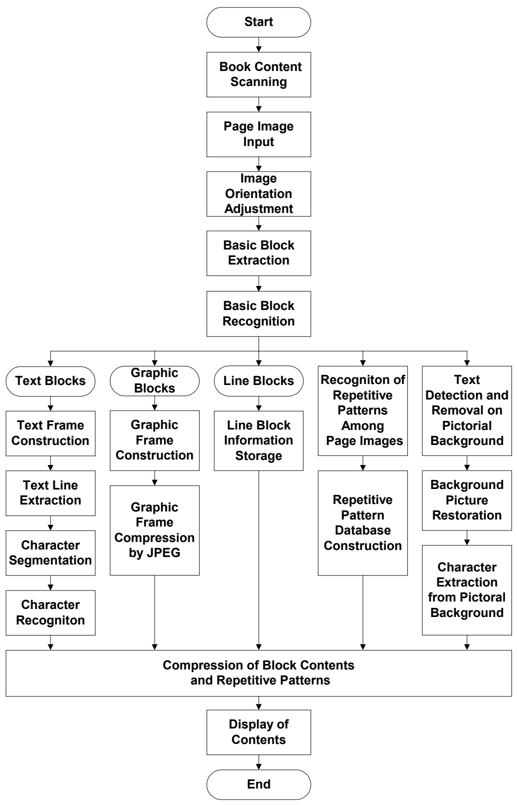

Start
Book Content Scanning
Page Image Input
Image Orientation Adjustment
Basic Block Extraction
Basic Block Recognition
Text Blocks
Graphic Blocks
Line Blocks
Recogniton of Repetitive Patterns Among Page Images
Text Detection and Removal on Pictorial Background
Text Frame Construction
Graphic Frame Construction
Line Block Information Storage
Text Line Extraction
Graphic Frame Compression by JPEG
Repetitive Pattern Database Construction
Background Picture Restoration
Character Segmentation
Character Extraction from Pictoral Background
Character Recogniton
Compression of Block Contents and Repetitive Patterns
Display of Contents
End

Fig.1. Flowchart of proposed system for electronic book construction and display.

The rest of this paper is organized as follows. In Section 2, some methods for extracting basic blocks of page images are proposed. In Section 3, some methods for classifying basic blocks and constructing text, graphic and outer frames are described. Moreover, the method for segmentation of characters in text frames is described. In Section 4, the proposed features and method for detecting repetitive patterns among page images are described. In Section 5, the proposed methods for detecting texts on pictorial backgrounds automatically are described. In Section 6, some methods proposed to organize and compress electronic book contents, and display them are described. Finally, some experimental results are shown in Section 7, and some conclusions are given in Section 8.

2 Automatic Basic Block Construction

2.1 Image Orientation Adjustment by Hough Transform

A page image obtained from scanning is sometimes skewed. It is necessary to adjust the orientation of the page image. The Hough transform is often used for detecting lines. The page images we want to process always consist of texts and graphics. Although they do not always include lines, but texts in the page image usually form several parallel-textured thick lines. Based on the use of this feature, we employ the Hough transform [6] to detect the skew angle of the page image. We then use a back transformation method to adjust the page image orientation. The advantage of using the back transformation is that it can avoid "hole" pixels in the created new image.

2.2 Meaningless Background Elimination by Moment-Preserving Thresholding

Thresholding is one of the most important approaches to image segmentation. The image thresholding technique adopted in this paper is based on an approach proposed by Tsai [5], which can automatically select threshold values by the moment-preserving principle. After executing the procedure, the meaningful contents are represented by black pixels. It will help us to devote our attention to process these meaningful contents of the page image.

2.3 Basic Block Segmentation by Region Growing and Merge by Geometric Position Analysis

In this paper, the region-growing algorithm [6] is used to construct basic blocks. After executing the procedure, the connected components of the page image are enclosed as rectangles. It helps us to devote our attention to processing these rectangles.

After segmenting the basic blocks, we get many blocks. Some of them overlap on another and some of them are very close to each other. Therefore, we merge these basic blocks into larger regions by closeness in their geometrical positions. The details are omitted here.

3 Automatic Image Content Segmentation

3.1 Features for Recognition of Basic Blocks

Basic blocks can be classified into text, graphic, and line blocks by their sizes and contents. Three size features are employed in this study:

1. the height of the basic block;
2. the width of the basic block;
3. the ratio of the height to the width or the ratio of the width to the height;

Based on our observations, if the size of a basic block is too large, the block may be a graphic block, and if the size of a basic block is too small, the block may be a noise block. If the ratio of the height to the width or the width to the height is too large, the block may be a line block.

The point density feature used in this study is as follows:

$$\text{Point density} = \frac{\text{the number of black pixels in a basic block}}{\text{the total number of pixels of the basic block}}. \qquad (1)$$

Based on our observations, the point densities of text blocks are higher than those of graphic blocks. Therefore, in this study if the point density of a basic block is higher than a threshold, this block is classified as a text block; otherwise, it is classified as a graphic block.

3.2 Classification of Text, Graphic, and Line Blocks

Algorithm 3.1 Classification of basic blocks.

Step 1: Classify an input basic block by the size features. If the ratio of the height to the width is smaller than a threshold t_1 or if the width to the height of the basic block is smaller than another threshold value t_2, go to Step 3. Otherwise, continue.

Step 2: If the width of the basic block is larger than a threshold t_3 *or* its height is larger than a threshold value t_4, then classify this block as a line block, go to Step 1, and continue classification with the next basic block as input. Otherwise, continue.

Step 3: Classify the basic block by the size features. If *both* the width *and* the length features of the block are smaller than a threshold value t_5, it is regarded as a noise block, go to Step 1, and continue classification with the next basic block as input. Otherwise, continue.

Step 4: Classify the basic block by the size features. If the width of the block is larger than threshold t_3 *and* its length is larger than threshold t_4, then it is regarded as a graphic block, go to Step 1, and continue classification with the next basic block as input. Otherwise, continue.

Step 5: Classify the basic block by the point density feature. If the value of the point density of the basic block is smaller than a threshold value t_6 *or* larger than another threshold value t_7, it is regarded as a graphic block, go to Step 1, and continue classification with the next basic block as input. Otherwise, continue.

Step 6: The block is regarded as a text block, go to Step 1, and continue classification with the next basic block as input.

The threshold value t_5 is selected by our experimental experience, and it is set to 1 pixel in this study. In the next section, a systematic method to calculate automatically the threshold values of t_1 through t_7 except t_5 is described.

3.3 Automatic Calculation of Classification Threshold Parameters

The threshold values we use in Algorithm 3.1 are the keys to good classifications of basic blocks. In general, these values are selected by experimental experiences after observing the page images. But it is time consuming to do so. Therefore, we propose systematic methods in this paper to calculate these threshold values automatically.

3.3.1 Calculation of size threshold parameters

The size features of a block include the height and the width of the block. According to our observations on the distributions of the widths and lengths of a lot of sample basic blocks in a width-length feature space shown in Fig. 2, a phenomenon of sample clustering is seen, that is, text blocks and noise blocks form a cluster and the other types of blocks on the contrary are spread all over the other part of the feature space. We found in this study that this phenomenon can be utilized to compute automatically some of threshold values used in Algorithm 3.1.

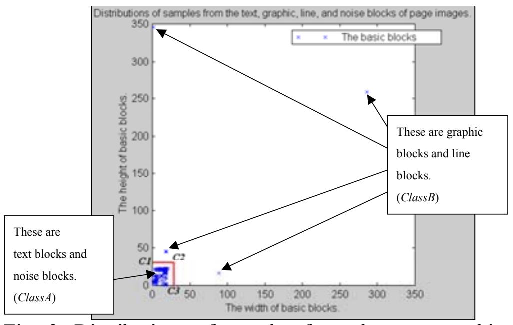


Fig. 2. Distributions of samples from the text, graphic, line, and noise blocks of page images.

More specifically, we design a classifier with the decision boundary of two line segments $\overline{C1C2}$ and $\overline{C2C3}$ to classify the basic blocks into *ClassA* containing text blocks and noise blocks; and *ClassB* containing graphic blocks and line blocks. Let the equations of the two line segments $\overline{C1C2}$ and $\overline{C2C3}$ be $y = y_0$ and $x = x_0$, respectively. Then a detailed comparison of the decision boundary and the threshold values t_1 through t_4 reveals that these threshold values are related to the values of x and y by the following formula:

$t_1 = 25 \times y_0 / x_0$;
$t_2 = 25 \times x_0 / y_0$;
$t_3 = x_0$;
$t_4 = y_0$.

Based on the above idea, we propose the following algorithm to calculate the classifier.

Algorithm 3.2 Automatic calculation of size threshold parameters.

ClassA: A class containing text blocks and noise blocks in the width-length feature space.

ClassB: A class containing graphic blocks and line blocks in the width-length feature space.

$Height_{avg}$: The average height of the basic blocks in *ClassA*.

$Width_{avg}$: The average width of the basic blocks in *ClassA*.

Step 1: Initially, assign all basic blocks to *ClassA*, and set $Height_{avg}$ and $Width_{avg}$ to zero.

Step 2: Calculate the average height and the average width of the blocks in *ClassA*. If the average height is equal to $Height_{avg}$ and the average width is equal to $Width_{avg}$, stop the iterations and go to Step 4. Otherwise, compute the new average height and the average width of all the blocks in *ClassA* and take them as the new values for $Height_{avg}$ and $Width_{avg}$, respectively, and then continue.

Step 3: Check all the basic blocks in the page image and discard those blocks whose heights and widths differ from $Height_{avg}$ and $Width_{avg}$, respectively, for a value larger than a threshold t_1. Go to Step 2.

Step 4: Take the values of $Height_{avg} + t_1$ and $Width_{avg} + t_1$ as the values for y_0 and x_0 mentioned previously, which may be taken to find the two segments $\overline{C1C2}$ and $\overline{C2C3}$, respectively, of the decision boundary of the classifier for assigning basic blocks into *ClassA* and *ClassB*.

In Step 3 and 4, the threshold t_1 is selected by experimental experience. Generally speaking, a book contains many fonts with different styles and sizes. But the difference in size between the biggest one and the smallest one does not exceed a range. So, t_1 is set to 10 pixels in this study.

An example of computing the size threshold parameters is shown in Fig. 3. The average height and average width were calculated three times before the iterations converged.

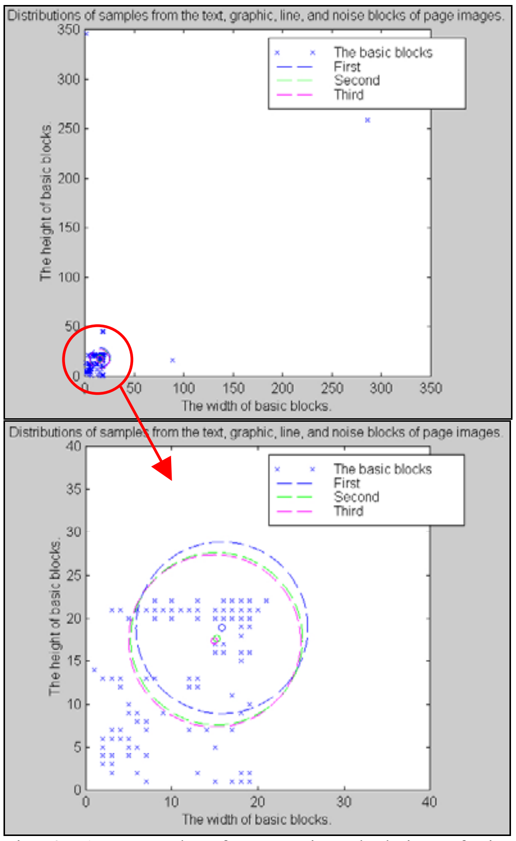


Fig. 3. An example of automatic calculation of size threshold parameters.

3.3.2 Calculation of point density threshold parameter

The point density of the basic block is another feature for classifying basic blocks. In general, text blocks have higher point density values than graphic ones. Line blocks have higher point density values than text blocks. So, we can design a classifier with two boundaries to classify the basic blocks into two classes. Let the two boundaries be specified by the two equations $y = y_1$ and $y = y_2$ with $y_1 < y_2$. Then a detailed investigation of Algorithm 3.1 reveals that the threshold values of t_6 and t_7 are related to the values of y_1 and y_2 by the following formula:

$$t_6 = y_2;\ t_7 = y_1.$$

Algorithm 3.3 Automatic calculation of point density threshold parameter.

UpperBound: The upper bound of the point density values of text blocks.

LowerBound: The lower bound of the point density values of text blocks.

ClassA: A class containing text blocks in the page image.

$PointDensity_{avg}$: The average point density of the basic block in *ClassA*.

Step 1: Initially, all basic blocks are regarded to belong to *ClassA*. And the values of $PointDensity_{\text{avg}}$, *UpperBound*, and *LowerBound* are set to be zero.

Step 2: Calculate the average point density of the blocks in *ClassA*. If the average point density is equal to $PointDensity_{\text{avg}}$, stop the iterations and go to Step 4. Otherwise, compute the new average point density of all the blocks in *ClassA* and take it as the new value of $PointDensity_{\text{avg}}$, and then go to Step 3.

Step 3: Check all the basic blocks in the page image and discard those blocks whose point density differs from $PointDensity_{\text{average}}$ for a value larger than a threshold t_1. Then go to Step 2.

Step 4: Take the values of $PointDensity_{\text{avg}} - t_1$ and $PointDensity_{\text{avg}} + t_1$ as the values of y_2 and y_1 mentioned previously, which can be used to find the decision boundaries *Upperbound* and *LowerBound*, respectively, of the classifier to classify basic blocks.

In Step 3 and Step 4, the threshold t_1 is selected by our experimental experience. An example of computing the point density threshold parameters is shown in Fig. 4.

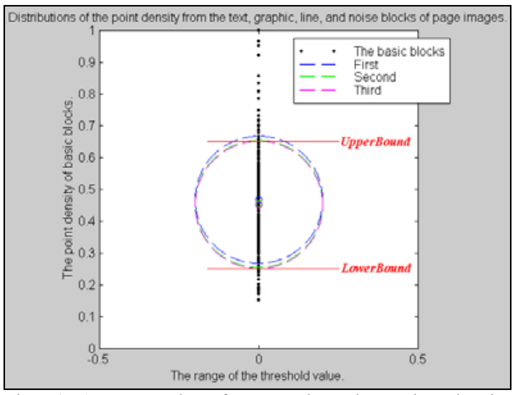


Fig. 4 An example of computing the point density threshold parameters.

3.4 Construction of Text, Graphic and Outer Frames

In order to process the page image content in more detail, we construct text, graphic and outer frames.

3.4.1 Graphic frame construction

A graphic block may have some related text blocks as its captions. These captions describe the content of the graphic. So we propose a method to select the text blocks that are the captions of graphic blocks and create a graphic frame to store the graphic block together with its captions.

3.4.2 Outer frame construction

In order to devote our attention to processing principal graphic and text blocks in the page image, the outer frame of a page need be segmented from the page image. Usually, the outer frame contains much information, including the page number, the footnote, the title of a chapter, the repetitive patterns, etc. An outer frame is constructed by four boundaries, *UpperBoundary*, *LowerBoundary*, *LeftBoundary* and *RightBoundary*. These boundary values are selected by experimental experience,

3.4.3 Text frame construction

We cannot gain much useful information from these small text blocks with incomplete knowledge of the contents of the page image. Therefore, how to merge these blocks into larger text frames is an important subject. We propose a method in this paper to merge text blocks into larger text frames.

3.5 Segmentation of Characters in Text Frames

Most OCR systems are applied to individual characters. Therefore we have to segment the characters in text frames. The projection method widely used for segmentation of characters is employed here.

3.5.1 Document orientation detection in text frames

In general, text frames have two text line directions: vertical and horizontal. The text line directions must be known first. Based on our observations, in a vertical text frame, the vertical projections can be divided more clearly; while in a horizontal text frame, the horizontal projections can be divided more clearly. We can use these properties to decide the orientations of text frames in page images.

VWhiteLength: the average length of gaps with zero values in the vertical projections.

HWhiteLength: the average length of gaps with zero values in the horizontal projections.

The decision rule based on the two types of average lengths is described as follows:

$$\begin{array}{l}\text{if VWhiteLength} > \text{HWhiteLength},\\ \text{decide the text frame to be vertical;}\\ \text{otherwise, horizontal.}\end{array} \qquad (5)$$

3.5.2 Text-line segmentation in text frames

The projection method can be used to segment text-lines in text frames further. The steps are described as follows, for the case of processing the vertical text frame.

Step 1: Apply the vertical projection method to get the vertical projection profiles of the text frame.

Step 2: Find the gaps in the profiles with zero projection values and record the boundary positions of the gaps.

Step 3: Use the middle points between the boundary positions to decompose the text frame into multiple text-lines.

The segmentation for the case of the

horizontal text frame can be accomplished with similar steps. We apply the horizontal projection method first, and execute the other two steps described above.

3.5.3 Character segmentation in text-line blocks

The projection function can be used to get the profiles of the text-lines. Using a threshold value for these profiles, the margins of the characters can be extracted. Some characteristics of Chinese characters like being of rectangular shapes are used. The steps are described as follows, for the case of processing the vertical text-lines.

Step 1: Apply the horizontal projection method to get the horizontal projection profiles of the text-line.

Step 2: Find the gaps of the horizontal projection profiles with zero values and record their positions. Set these positions as the margins of characters.

Step 3: Select the first margin from top to bottom in the text-line as the base margin.

Step 4: Discard those margins whose positions differ from the position of the base margin by a distance smaller than a threshold t_1.

Step 5: If all margins of the characters have been processed, go to Step 6. Otherwise, select the next margin as the base margin, and go to Step 4.

Step 6: Record the remaining margins and use them as boundaries to segment the characters.

The segmentation for the case of processing horizontal text-lines can be accomplished by similar steps. We apply the vertical projection method first, and execute the steps described above.

4 Automatic Repetitive Pattern Detection in Pages

4.1 Features for Detection of Repetitive Patterns

In this section, some proposed features that are used to detect the repetitive patterns among different pages in a book are described.

4.1.1 Size feature

The size of the block or frame is the main properties to detect repetitive patterns. It is impossible to decide two blocks with different sizes as repetitive patterns. Two size features of blocks or frames are used in this paper:

1. the height of the block or frame;
2. the width of the block or frame.

Based on our observations, similar parts among pages have similar size features. Therefore, if the heights and widths of two blocks or frames in different page images are similar, these two blocks or frames may be repetitive patterns.

4.1.2 Block content feature

The content feature is not an absolute necessary feature for deciding repetitive patterns. The page number is an example of this phenomenon. The page numbers contain similar block sizes, but their contents are different actually. Therefore, we can detect and classify the repetitive patterns by the content feature as follows.

Full repetitive pattern: the patterns, which contain similar block sizes and identical contents.

Semi-repetitive pattern: the patterns, which only contain similar block sizes but different contents.

4.1.3 Continuity feature

Based on our observations, most repetitive patterns appear in each consecutive page image. However, the repetitive patterns may sometimes appear only in odd page images or only in even page images. Therefore, the continuity of the repetitive pattern is a feature to detect and classify repetitive patterns.

Continuous repetitive pattern: the patterns, which appear in continuous page images.

Skipping repetitive pattern: the patterns, which appear only in odd page images or even page images.

4.1.4 Geometric position feature

In general, the repetitive patterns between two pages always have similar geometric positions. But sometimes the repetitive patterns might have symmetrical geometric positions between two pages. Therefore, the geometric position of the repetitive pattern is a feature to detect and classify the repetitive patterns.

Identical-positioned repetitive pattern: The repetitive patterns have similar geometric positions.

Reflective repetitive pattern: The repetitive patterns have symmetrical geometric positions.

4.2 Classification of Repetitive Patterns

The features for classification of repetitive patterns we described above are all independent. Therefore, we can combine any three types of them to form a new type of repetitive pattern.

Type 1: Continuous identical-positioned full repetitive pattern.

The repetitive patterns in this type contain similar block sizes, geometric positions, and identical contents. The repetitive patterns appear in continuous page images. Fig. 5 shows an example of continuous identical-positioned full repetitive pattern.

Type 2: Continuous reflective full repetitive pattern.

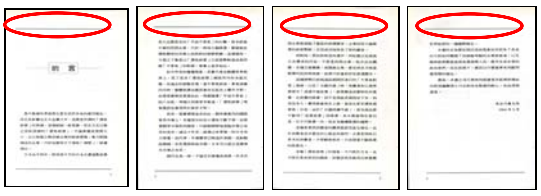


Fig. 5 Continuous identical-positioned full repetitive pattern.

The repetitive patterns of this type contain similar block sizes, symmetrical positions, and identical contents. The repetitive patterns appear in continuous page images. Fig. 6 shows an example of continuous reflective full repetitive pattern.

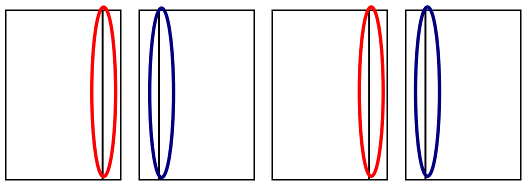


Fig. 6 Continuous reflective full repetitive pattern.

Type 3: Skipping identical-positioned full repetitive pattern.

The repetitive patterns in this type contain similar block sizes, geometric positions, and identical contents. The repetitive patterns only appear in odd page images or even page images. Fig. 7 shows an example of skipping identical-positioned full repetitive pattern.

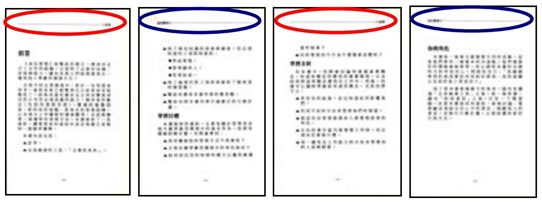


Fig. 7 Skipping identical-positioned full repetitive pattern.

Type 4: Continuous identical-positioned semi-repetitive pattern.

The repetitive patterns of this type contain similar block sizes and geometric positions. The repetitive patterns appear in continuous page images. But the pattern contents are not identical. Fig. 8 shows an example of continuous identical-positioned semi-repetitive pattern.

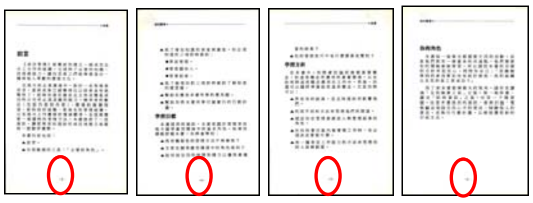


Fig. 8 Continuous identical-positioned semi-repetitive pattern.

Type 5: Continuous reflective semi-repetitive pattern.

The repetitive patterns of this type contain similar block sizes and symmetrical positions. The repetitive patterns appear in continuous page images. But the pattern contents are not identical. Fig. 9 shows an example of continuous reflective semi-repetitive pattern.

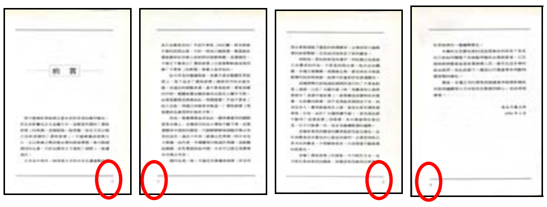


Fig. 9 Continuous reflective semi-repetitive pattern.

Type 6: Skipping identical-positioned semi-repetitive pattern.

The repetitive patterns of this type contain similar block sizes and geometric positions. The repetitive patterns only appear in odd page images or even page images. But the pattern contents are not identical. Fig. 10 shows an example of skipping identical-positioned semi-repetitive pattern.

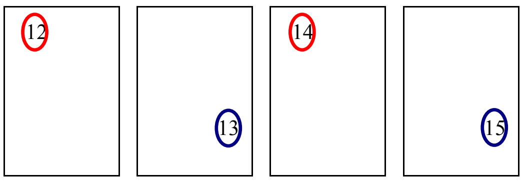


Fig. 10 Skipping identical-positioned semi-repetitive pattern.

Type 7: Skipping reflective full repetitive pattern.

The repetitive patterns of this type contain similar block sizes, symmetrical positions, and identical contents. The repetitive patterns only appear in odd page images or even page images. Based on our observations, we seldom see books with this type of repetitive pattern. So this type is not considered in our paper.

Type 8: Skipping reflective semi-repetitive pattern.

The repetitive patterns of this type contain similar block sizes and symmetrical positions. The repetitive patterns only appear in odd page images or even page images. But the pattern contents are not identical. Based on our observations, we seldom see books with this type of the repetitive pattern. So this type is not considered in our paper.

4.3 Hierarchical Repetitive Pattern Detection

A hierarchical method is proposed to detect repetitive patterns among different pages in a book. This method includes two phases to detect repetitive patterns. In the first phase, we detect repetitive patterns based on the size, geometric position, and content features. In the second phase, we check the continuity of the repetitive patterns we have found in the first phase.

Algorithm 4.1 Detection of repetitive patterns among different pages in a book.

Phase 1:

Step 1: Input all page images of the book. Each page image must be compared with two other page images: one is the next page image, and the other is the page image after the next page image. We start at the first page image of the book.

Step 2: Define the first page image of the book as the base image and denote it as *Base*, and denote the next page image of the base image as *Target*.

Step 3: Get a block B_1 from *Base*, and a block B_2 from *Target*.

Step 4: Compare B_1 and B_2 by the size features. If the sizes of the blocks are similar, go to Step 5. Otherwise, go to Step 3 and continue with the next block in *Target*.

Step 5: Compare B_1 and B_2 by the content feature using template matching. If the content features of these two blocks are similar, then decide B_1 and B_2 as full repetitive patterns and go to Step 6. Otherwise, decide B_1 and B_2 as semi-repetitive patterns and go to Step 7.

Step 6: Compare B_1 and B_2 by the geometric position feature. If the geometric positions of the two blocks are similar, then decide B_1 and B_2 as identical-positioned full repetitive patterns. Otherwise, if the two blocks are at reflective positions, decide B_1 and B_2 as reflective full repetitive patterns. Go to Step 8.

Step 7: Compare B_1 and B_2 by the geometric position feature. If the geometric positions of the two blocks are similar, then decide B_1 and B_2 as identical-positioned semi-repetitive patterns. Otherwise, if the two blocks are at reflective positions, decide B_1 and B_2 as reflective semi-repetitive patterns. Go to Step 8.

Step 8: Repeat Steps of 3 through 7, until all blocks in *Base* and *Target* are processed in pairs.

Step 9: Define the page image after the next page image as *Target*, and repeat Steps of 3 through 8.

Step 10: Redefine the next page image of the base image as *Base*, and repeat steps of 2 through 9, until all page images of the book are processed.

In Step 4 and Step 6, several threshold values are used to describe the similarity between the blocks. These threshold values are selected by our experimental experience, and they are set to 5 pixels in this study.

In Step 5, a block matching method is used to measure the matching degree of the blocks.

Phase 2:

In this phase, we classify the repetitive patterns based on the continuity feature.

Step 1: Input all page images of the book. Each page image is compared only with the next page image. We start at the first page image of the book.

Step 2: Define the first page image of the book as the base image and denote it as *Base*, and denote the next page image of the base image as *Target*.

Step 3: Get an identical-positioned full repetitive pattern R_1 from *Base*, and an identical-positioned full repetitive pattern R_2 from *Target*.

Step 4: Compare R_1 and R_2 by the geometric position feature. If the geometric positions of R_1 and R_2 are similar, then decide R_1 and R_2 as continuous identical-positioned full repetitive patterns. Otherwise, decide R_1 and R_2 as skipping identical-positioned full repetitive patterns. Go to Step 3, and continue processing next identical-positioned full repetitive patterns in *Base* and *Target*, until all identical-positioned full repetitive patterns are processed, then go to Step 5.

Step 5: Get an identical-positioned semi-repetitive pattern R_3 from *Base*, and an identical-positioned semi-repetitive pattern R_4 from *Target*.

Step 6: Compare R_3 and R_2 by the geometric position feature. If the geometric positions of R_3 and R_4 are similar, then decide R_3 and R_4 as continuous identical-positioned semi-repetitive patterns. Otherwise, decide R_3 and R_4 as skipping identical-positioned semi-repetitive patterns. Go to Step 5, and continue processing next identical-positioned semi-repetitive patterns in *Base* and *Target*, until all identical-positioned semi-repetitive patterns are processed, then go to Step 7.

Step 7: Get a reflective full repetitive pattern R_5 from *Base*, and a reflective full repetitive pattern R_6 from *Target*. And decide R_5 and R_6 as continuous reflective full repetitive patterns. Continue until all reflective full repetitive patterns are processed, and then go to Step 8.

Step 8: Get a reflective semi-repetitive pattern R_7 from *Base*, and a reflective semi-repetitive pattern R_8 from *Target*. And decide R_7 and R_8 as continuous reflective semi-repetitive patterns. Continue until all reflective semi-repetitive patterns are processed, and then go to Step 9.

Step 9: Redefine the next page image of the base image as *Base*, and repeat steps of 2 through 8, until all page images of the book are processed.

In Step 7 and Step 8, we ignore the skipping reflective full repetitive pattern and the skipping reflective semi-repetitive pattern. Therefore, we can classify the reflective full repetitive pattern into the continuous reflective full repetitive pattern and the reflective semi-repetitive pattern into the continuous reflective semi-repetitive pattern immediately.

After executing the algorithm, repetitive patterns with different types are detected and classified into six types. We show some experimental results of detecting repetitive patterns between page images in a book in Fig. 11(a) and Fig. 11(b).

5 Automatic Text Detection on Pictorial Background

5.1 Text Detection and Removal by Dilation

There often exist texts on the pictorial background to describe the contents of the pictorial background. Detection and analysis of these texts helps gain much useful information. Therefore we propose an algorithm to detect and analyze these texts.

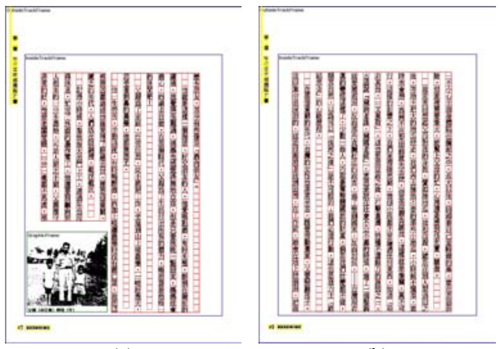

(a) (b)

Fig. 11. An example of experimental results of hierarchical repetitive pattern detection. (a) One page image in a book. (b) Another page image in a book. The yellow rectangles in the page images (a) and (b) are the skipping identical-positioned full repetitive patterns.

Algorithm 5.1 Text detection and removal on pictorial background.

Step 1: Input a page image including a pictorial background and apply the Sobel operation to each of the RGB planes, respectively. If the average gray values of the RGB planes at a pixel are larger than a threshold t_1, this pixel is taken as an edge point.

Step 2: Use the region-growing algorithm to construct basic blocks from the edge points found in Step 1. The details are described in Section 2.3.

Step 3: Classify basic blocks into text and graphic blocks by the size and point density features of the blocks that we described in Section 3.2.

Step 4: After basic blocks are classified, pay attention to the text blocks. Analyze the geometric position of the text blocks and merge the text blocks into more complete text blocks. Record the positions of these text blocks. The merge method is described in Section 2.4.

Step 5: Reload the original page image. Transform all pixels in the text blocks from the RGB to the YIQ color model.

Step 6: Get a pixel P in the text block and check other pixels in the neighborhood of P (the neighborhood of each pixel is defined to be a 3 × 3 region with the pixel as the center) to find the pixel P' with the maximum Y values.

Step 7: Replace the R, G, and B color values of pixel P with the R, G, and B color values of P'.

Step 8: Repeat Step 6 to Step 7 until all pixels in the text block are processed.

In Step 1, the threshold t_1 is selected by the moment-preserving method we described in Section 2.2. In Step 5, we use the YIQ color model [6] in Algorithm 5.1. An advantage to use the YIQ color model is that the luminance (Y) and color information (I and Q) are decomposed. Thus the color information (I and Q) is preserved. An example of the results of text detection and removal is shown in Fig. 12(a)(b).

5.2 Background Picture Restoration

In this section, we propose a method to restore the contents on the pictorial background.

Algorithm 5.2 Background picture restoration.

Step 1: Input the picture produced by text detection and removal in Section 5.1. Initially, define ΔR, ΔG and ΔB to denote the average of the color differences between the text blocks and an outer track of the text blocks on the RGB planes.

Step 2: Compute the average of the color differences between the text blocks and the outside track of the text blocks on the RGB planes as:

$$\Delta R = \frac{\sum_{(x,y)\in IN} R(x,y)}{N_{IN}} - \frac{\sum_{(x,y)\in OUT} R(x,y)}{N_{OUT}},$$
$$\Delta G = \frac{\sum_{(x,y)\in IN} G(x,y)}{N_{IN}} - \frac{\sum_{(x,y)\in OUT} G(x,y)}{N_{OUT}}, \quad (2)$$
$$\Delta B = \frac{\sum_{(x,y)\in IN} B(x,y)}{N_{IN}} - \frac{\sum_{(x,y)\in OUT} B(x,y)}{N_{OUT}},$$

where IN denotes the region of the text blocks, and OUT denotes the outer track region of the text blocks; N_{IN} denotes the numbers of the pixels in the text blocks; N_{OUT} denotes the numbers of the pixels in the outer track of the text blocks; and $R(x,$

y), $G(x, y)$ and $B(x, y)$ are the three-channel color values at the position (x, y) on the RGB planes.

Step 3: Finally, increase the three-channel color values of each pixel in the text blocks on the RGB planes by the amounts of ΔR, ΔG and ΔB, respectively, as follows:

$$\begin{aligned} R_{IN}(x,y) &= R_{IN}(x,y) + \Delta R \quad \forall (x,y) \in IN, \\ G_{IN}(x,y) &= G_{IN}(x,y) + \Delta G \quad \forall (x,y) \in IN, \quad (3) \\ B_{IN}(x,y) &= B_{IN}(x,y) + \Delta B \quad \forall (x,y) \in IN. \end{aligned}$$

In Step 1, the thickness of the outer track of the text blocks is selected by our experimental experiences, and is set to 5 pixels in this paper.

An example of the results produced by the proposed method for background picture restoration is shown in Fig. 12(b)(c).

5.3 Character Extraction from Pictorial Background

After restoring the background picture, we get a pure picture without any text. However, the texts on the pictorial background contain much useful information. Therefore, we propose a method to extract characters from the pictorial background.

Algorithm 5.3 Extraction of Characters from pictorial background.

Image$_{\text{original}}$: the original page image.

Image$_{\text{pure}}$: the page image after removing the text blocks and restoring the background picture obtained by the method described in Section 5.2.

Image$_{\text{difference}}$: the result page image after extracting characters from the pictorial background.

Step 1: Transform *Image*$_{\text{original}}$ and *Image*$_{\text{pure}}$ into gray-level images.

Step 2: Thus, for each pixel at row x and column y in the *Image*$_{\text{original}}$, *Image*$_{\text{pure}}$, and *Image*$_{\text{difference}}$, we compute:

$$\begin{aligned} R(x,y)_{difference} &= R(x,y)_{original} - R(x,y)_{pure}, \\ G(x,y)_{difference} &= G(x,y)_{original} - G(x,y)_{pure}, \quad (4) \\ B(x,y)_{difference} &= B(x,y)_{original} - B(x,y)_{pure}, \end{aligned}$$

where *original* denotes the original page image, *pure* denotes the page image after restoring the background picture and *difference* denotes the resulting page image.

Step 3: Construct the page image from the color values of $R_{difference}$, $G_{difference}$, and $B_{difference}$ as the result of extracting characters from the pictorial background.

An example of the results produced by the proposed method for character extraction from pictorial background is shown in Fig. 12(d).

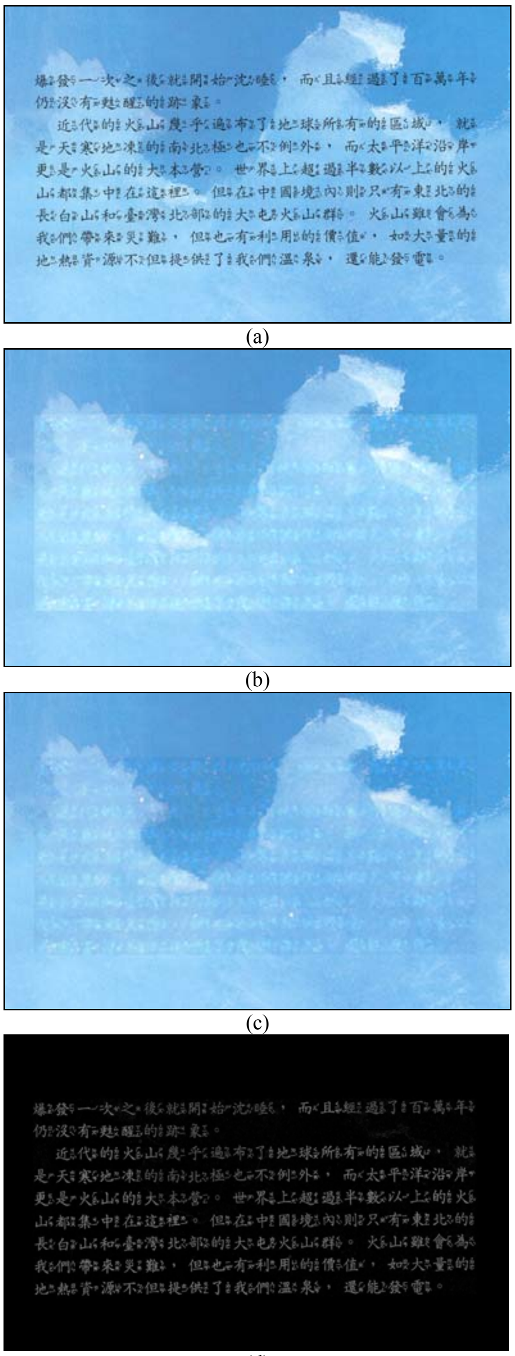


Fig. 12. An example of the results of automatic text detection on the pictorial background. (a) The original page image. (b) The page image after detecting and removing texts. (c) The page image after restoring background picture. (d) The page image after extracting characters from the pictorial background.

6 Automatic Page Content Reconstruction

6.1 Characters Recognition

In Section 3.5, a character data structure is used to store the information of characters. Then, we start to recognize these characters. First, we construct a gray level image for each character. Then, we apply an OCR system to recognize these gray level images. Finally, we save the recognition results in the text file format.

6.2 Compression of Graphic Frames

JPEG[7][8] is a standard for compressing images. Because of the high compression rate of the JPEG compression algorithm, we apply it in this study to compress the graphic frame. But the captions of the graphic in the graphic frame must be recognized first. For this, we use the OCR process described previously and save the result as part of the graphic frame data.

6.3 Compression of Repetitive Patterns

Each repetitive pattern in page images is a pattern block, which belongs to one of the types of text, graphic, and line blocks. Use of different compression techniques for different types of blocks can increase the overall compression rate of the electronic book. Use the OCR system to recognize all of the characters of each text block. Apply the JPEG compression algorithm to compress each graphic block. Compute and save the position and the thickness of each line block. These blocks belong to the repetitive patterns.

6.4 Electronic Book Organization and Compression

Now, each component in the page image is compressed. In order to reconstruct and display the electronic book, the attributes of these components in the page images must be stored. We construct a file to save the attributes of these components in the page images.

1. **Graphic block attributes:** Graphic path and Graphic position.
2. **Text block attributes:** Text content and Text position.
3. **Line block attributes:** Line thickness, Line start position and Line end position
4. **Repetitive pattern attributes:** Repetitive pattern position, Repetitive pattern index and Repetitive pattern type.
5. **Page attributes:** Page image file path, Page width, Page height, Line number, Graphic number, and Text number.
6. **Repetitive pattern database attributes:** Line number, Text number and Graphic number.
7. **Electronic book attributes:** Pagination.

These attributes in the INI file form a tree structure based on the data types of the components. We construct the electronic book according to these attributes of components in the page images.

Algorithm 6.1. Electronic book construction.

```
Step 1: Define a book container to save the
        components of the page images of a book.
Step 2: Input the first page image of the book.
        Save the attributes of the text, graphic,
        and line blocks of this page image in a file
        and store these components in the
        electronic book container. Save the
        attributes of the repetitive patterns of this
        page image in the file and store the
        repetitive patterns according to the types
        of the repetitive patterns in the repetitive
        pattern database, respectively.
Step 3: Input the second page image of the book.
        Save the attributes of text, graphic, and
        line blocks of this page image in the file
        and store these components in the
        electronic book container.
Step 4: Get a repetitive pattern $R$ of the second
        page image. Compare $R$ and the elements
        of the repetitive pattern database. If there
        exists an element $RP_1$ in the repetitive
        pattern database with its type and
        attributes all similar to $R$, point $R$ toward
        $RP_1$ and save the displacement between $R$
        and $RP_1$ in the repetitive pattern database.
Step 5: Repeat Steps of 3 through 4, until all
        page images are processed.
```

The geometric positions of the repetitive patterns between page images always have a little difference. Therefore, in Step 4, we must store the displacement between the blocks and the element in the repetitive pattern database.

After implementing Algorithm 6.1, the page images has been organized and compressed into an electronic book. An illustration of the proposed organization of the data structure for the electronic book is shown in Fig. 13. This tree structure can help us to reconstruct the original electronic book easily.

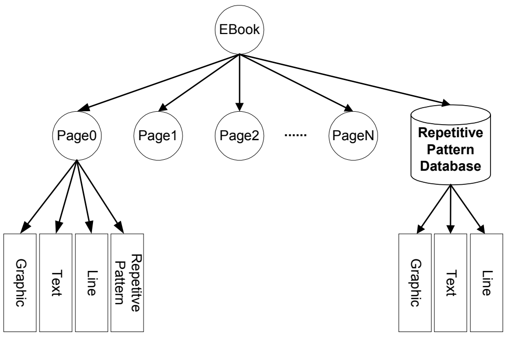


Fig. 13 A simple example of organizing the electronic book.

7 Experimental Results

Some page images of a book were tested in our experiment. We obtain these page images from an automatic document feeder of the HP ScanJet scanner at 150dpi resolution with true color levels. The proposed system was implemented on a Pentium III-933 PC with 256 MB RAM and software development was conducted by the use of VC++ 6.0 in a Windows 2000 Professional platform.

The result of the electronic book is shown in Fig. 14. Fig. 14(a) shows the source image, Fig. 14(b) shows the result of segmentation and classification, and Fig. 14(c) shows the result of reconstruct the electronic book.

8 Conclusions

A system for automatic construction of electronic book contents from scanned book page images has proposed. In the phase of analysis of page image contents, a method was proposed to classify page contents into different types of blocks by the size and point density features. Furthermore, methods to calculate the size and point density threshold parameters automatically were proposed. In the phase of detection of repetitive patterns in image pages, a hierarchical method was proposed. The detection results can be utilized to improve the book content compression rate. In the phase of removal of texts on the pictorial background, we proposed a method to remove these texts. Furthermore, a method was proposed to extract characters on the pictorial background using the pure picture obtained from removing the texts as well as the original picture. In the phase of compression of page components, a method was proposed to construct a repetitive pattern database to improve the compression rate further. Finally, a complete data structure of a book was designed to reconstruct the electronic book for display and reading. The experimental results have revealed the feasibility of the proposed methods.

References

[1] C.J. Park, J.H. Jeon, T.M. Koo, and H.M. Choi, "An Edge-Based Block Segmentation and Classification for Document Analysis with Automatic Character String Extraction," *Proceedings of IEEE on International Conference Systems, Man and Cybernetics*, Vol.1, pp.707-712, 1996.

[2] J. H. Bae, L. C. Jung, J. W. Kim, H. J. Kim, "Segmentation of Touching Character Using an MLP," *Pattern Recognition Letters*, vol. 19, pp. 701-709, 1998.

[3] S. H. Chen and W. H. Tsai, "Book Content Digitization and Display for Digital Library by Document Image Analysis and Compression-By-Classification Techniques", *Proc. of 2000 IPPR Conf. On CVGIP*, Taipei, Taiwan, ROC, pp. 23-32, 2000.

[4] W. H. Tsai and C. Y. Chan, "A Bottom-Up Approach to Color Image Document Analysis and Rearrangement," *Department of Computer and Information Science, National Chiao Tung University*, pp. 7-42, June 1999.

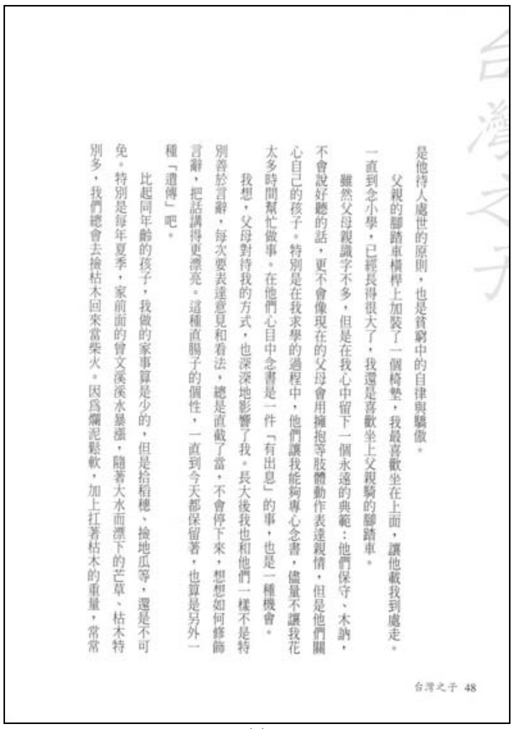
[5] W. H. Tsai, "Moment-preserving thresholding: a new approach," *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 377-393, 1985.

[6] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley Publishing Company, U.S.A., 1993.

[7] G. K. Wallace, "The JPEG Still Picture Compression Standard," *IEEE Transactions on Consumer Electronics*, Vol. 38, No. 1, February 1992.

[8] JPEG. Digital compression and coding of continuous tone still images - requirements and guidelines. ITU recommendation T.81, ISO/IEC International Standard 10918 - 1,1993.

[9] Y. C. Yang and C. S. Fuh, " Chinese Character Segmentation in Machine Printed Documents," in 7th Optical Character Recognition and Document Analysis Workshop, November 1997, pp. 2.20-2.23.



台灣之子

是他待人處世的原則，也是貧窮中的自律與驕傲。

父親的腳踏車橫桿上加裝了一個椅墊，我最喜歡坐在上面，讓他載我到處走。一直到念小學，已經長得很大了，我還是喜歡坐上父親騎的腳踏車。

雖然父母親識字不多，但是在我心中留下一個永遠的典範：他們保守、木訥，不會說好聽的話，更不會像現在的父母會用擁抱等肢體動作表達親情，但是他們關心自己的孩子。特別是在我求學的過程中，他們讓我能夠專心念書，儘量不讓我花太多時間幫忙做事。在他們心目中念書是一件「有出息」的事，也是一種機會。

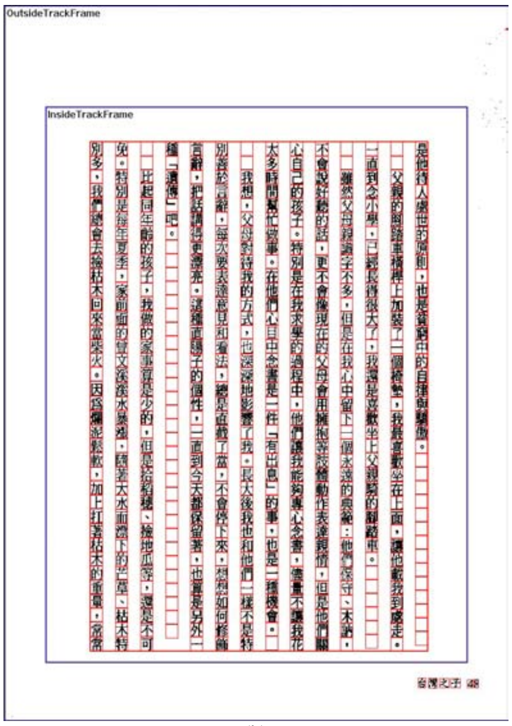
我想，父母對待我的方式，也深深地影響了我。長大後我也和他們一樣不是特別善於言辭，每次要表達意見和看法，總是直截了當，不會停下來，想想如何修飾言辭，把話講得更漂亮。這種直腸子的個性，一直到今天都保留著，也算是另外一種「遺傳」吧。

比起同年齡的孩子，我做的家事算是少的，但是拾稻穗、撿地瓜等，還是不可免。特別是每年夏季，家前面的曾文溪溪水暴漲，隨著大水而漂下的芒草、枯木特別多，我們總會去撿枯木回來當柴火。因爲爛泥鬆軟，加上扛著枯木的重量，常常

台灣之子 48

(a)

Fig. 14. Experimental results. (a) The source image. (b) The resulting image after document segmentation. (c) The electronic book page.

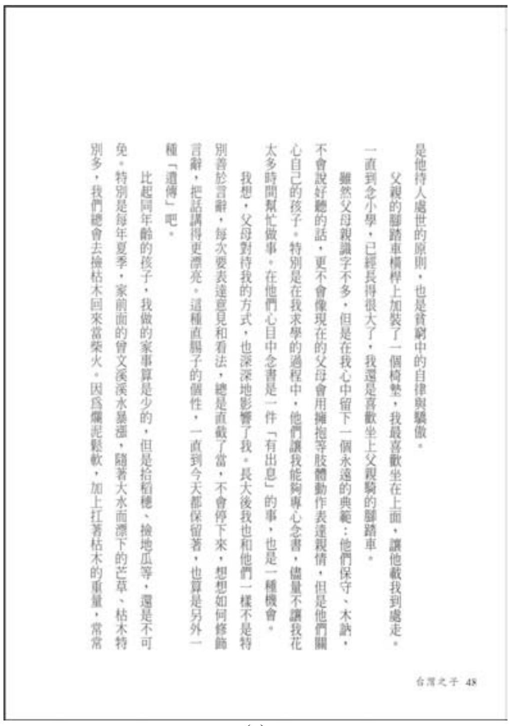

OutsideTrackFrame
InsideTrackFrame

是他待人處世的原則，也是貧窮中的自律與驕傲。
　父親的腳踏車橫桿上加裝了一個椅墊，我最喜歡坐在上面，讓他載我到處走。
一直到念小學，已經長得很大了，我還是喜歡坐上父親騎的腳踏車。
　雖然父母親識字不多，但是在我心中留下一個永遠的典範：他們保守、木訥，
不會說好聽的話，更不會像現在的父母會用擁抱等肢體動作表達親情，但是他們關
心自己的孩子。特別是在我求學的過程中，他們讓我能夠專心念書，儘量不讓我花
太多時間幫忙做事。在他們心目中念書是一件「有出息」的事，也是一種機會。
　我想，父母對待我的方式，也深深地影響了我。長大後我也和他們一樣不是特
別善於言辭，每次要表達意見和看法，總是直截了當，不會停下來，想想如何修飾
言辭，把話講得更漂亮。這種直腸子的個性，一直到今天都保留著，也算是另外一
種「遺傳」吧。
　比起同年齡的孩子，我做的家事算是少的，但是拾稻穗、撿地瓜等，還是不可
免。特別是每年夏季，家前面的曾文溪溪水暴漲，隨著大水而漂下的芒草、枯木特
別多，我們總會去撿枯木回來當柴火。因爲爛泥鬆軟，加上扛著枯木的重量，常常

台灣之子 48

(b)

Fig. 14. Experimental results. (a) The source image. (b) The resulting image after document segmentation. (c) The electronic book page (continued).



是他待人處世的原則，也是貧窮中的自律與驕傲。
　父親的腳踏車橫桿上加裝了一個椅墊，我最喜歡坐在上面，讓他載我到處走。
一直到念小學，已經長得很大了，我還是喜歡坐上父親騎的腳踏車。
　雖然父母親識字不多，但是在我心中留下一個永遠的典範：他們保守、木訥，
不會說好聽的話，更不會像現在的父母會用擁抱等肢體動作表達親情，但是他們關
心自己的孩子。特別是在我求學的過程中，他們讓我能夠專心念書，儘量不讓我花
太多時間幫忙做事。在他們心目中念書是一件「有出息」的事，也是一種機會。
　我想，父母對待我的方式，也深深地影響了我。長大後我也和他們一樣不是特
別善於言辭，每次要表達意見和看法，總是直截了當，不會停下來，想想如何修飾
言辭，把話講得更漂亮。這種直腸子的個性，一直到今天都保留著，也算是另外一
種「遺傳」吧。
　比起同年齡的孩子，我做的家事算是少的，但是拾稻穗、撿地瓜等，還是不可
免。特別是每年夏季，家前面的曾文溪溪水暴漲，隨著大水而漂下的芒草、枯木特
別多，我們總會去撿枯木回來當柴火。因爲爛泥鬆軟，加上扛著枯木的重量，常常

台灣之子 48

(c)

Fig. 14. Experimental results. (a) The source image. (b) The resulting image after document segmentation. (c) The electronic book page (continued).