Image Processing for Historical Newspaper Archives

Takahiro Shima
Renesas Micro Systems Co., Ltd.
1-1 Nishi 7-chome, Kita 1-jo, Chuo ward, Sapporo,
Hokkaido, 060-0001 Japan
takahiro.shima.jg@rms.renesas.com

Kengo Terasawa and Toshio Kawashima
Graduate School of Systems Information Science,
Future University Hakodate
116–2 Kamedanakano, Hakodate, Hokkaido, 041–8655
Japan
{kterasaw,kawasima}@fun.ac.jp

***Abstract*— This paper presents some image processing methods that could produce accurate character segmentation results for historical newspaper archives. A full text search using a word spotting technique is no doubt a promising approach in order to facilitate the utilization of digital archives. Some word spotting techniques require the target images to be segmented into character images in advance, however character segmentation is a difficult issue especially for old and degraded document images. This paper figures out the causes that make the character segmentation difficult, and removes them in order to improve the accuracy of character segmentation. We first detect the ruled lines using Hough Transform in order to segment a whole newspaper image into column-separated images. Then we remove the ruled lines as well as ruby characters and noise. The proposed system is tested for 20 column-separated images of historical newspapers, and the accuracy of character segmentation is improved to 96.3%.**

***Keywords*-historical document; full text search; character segmentation; optical character recognition; digital archive**

I. INTRODUCTION

Recently, preservation of historical documents as digital archives becomes an important issue for historical study. To exhibit historical documents archived as printed books is more difficult than with photo or picture materials because viewers need to touch the materials if they want to read the whole document, but this would be prohibited from the viewpoint of material protection. Hence it is valuable to browse historical document archives with a computer in digital image format. Examples of such web pages that display digital archives are The National Institute of Japanese Literature [1], The National Diet Library [2], and so on.

However, as the amount of archived document images grows, searching intended information from the archives is getting more difficult. One way to resolve this problem is to apply optical character recognition (OCR) software and use string searching to find intended information. It seems to be a good idea because the printed documents' layout is considered usually in good order and applying OCR to them seems to be easy. In fact, however, off-the-shelf OCR software cannot provide satisfactory results since historical newspapers are different from recent newspapers in image quality, type fonts, ruby characters, noise, and language usage. Correcting errors of such OCR results requires an enormous effort of professional researchers, and it is difficult in practice. In order to effectively utilize document digital archives, improving the image processing methods for old and degraded document images is indispensable. We propose some image processing methods that could improve the accuracy of character segmentation, which is the first important process to deal with in historical document images.

A. Target material and dataset

The target material in this study is the 1878 to 1884 issues of the “Hakodate Shimbun,” a Japanese local old newspaper published from 1878–1908. This is archived as microfilm in the Hakodate Central Library, Japan. The archive consists of 3,462 pages. The archive was digitized into image format using an image scanner. We use these image files as our dataset. The image files are formatted as binary TIFF format with a size of 6072x8600 pixels. The file size of each image is approximately 1.83MB.

In the layout of our material "Hakodate Shimbun", there is a thick outer frame surrounding a body text area and there are ruled lines segmenting a whole body text area into column-separated areas. Most pages have a vertical three-column format except for the masthead region (Fig.1). The body text areas still contain some hindrances for character segmentation such as ruby characters and noise. There are a lot of ruby characters to the right of most kanji characters. There is also much noise caused by show-through and sanction seal. Considering these attributes of the layout, the proposed image processing and character segmentation processes are constructed.

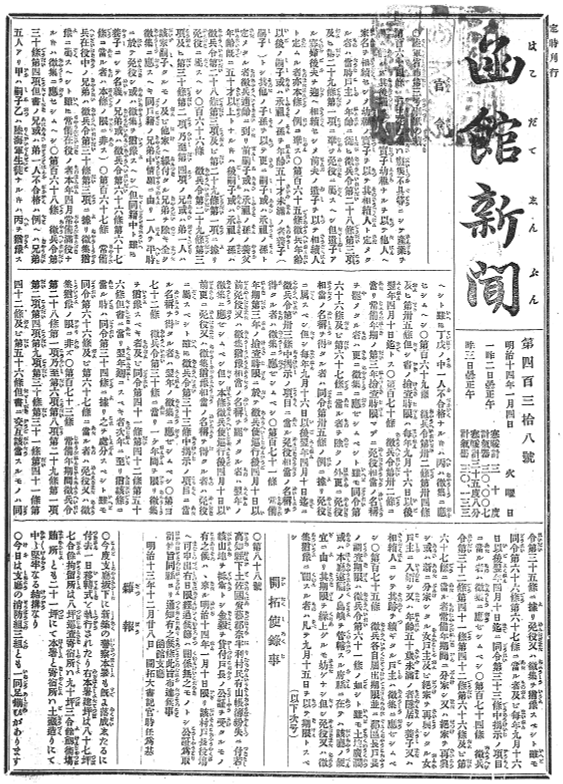
B. Our previous studies

We previously researched a fast full text search method for historical handwritten documents [3,4]. It utilized not OCR, but a word spotting technique.

HIP'11, September 16 - September 17 2011, Beijing, China

We have tested the same method for the historical newspaper images, and the searching performance was satisfactory. In using this method, however, from the viewpoint of computational cost, it is preferable to segment newspaper images into character images in advance, and this was a difficult issue. In the early period of the study, we did the segmentation using the projection method and manual correction. In fact this work was a bit hard because the projection method did not produce such a good result that the workload for manual correction was heavy. Next we tried to use off-the-shelf OCR software to obtain a character segmentation result, but not to obtain a character recognition result. This trial was based on our expectation that even though off-the-shelf OCR software was not able to produce good results for character recognition, it might produce good results for character segmentation. However, the result did not meet our expectations. Figure 2 represents the character segmentation result for "Hakodate Shimbun" using off-the-shelf OCR software. Apparently the accuracy of character segmentation was low. We still needed to pay an enormous effort to correct the result manually.

From the experience of such manual correction operations, we have found some typical problems that worsen the character segmentation result. If these problems can be solved, we expect, the accuracy of character segmentation will be improved and that will make a highly accurate full-text search with [3,4] feasible for a large-sized dataset.



函館新聞

Figure 1. A typical page of our material: “Hakodate Shinbun.”

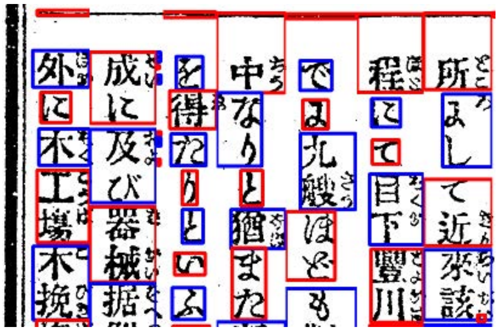


Figure 2. An example of the character segmentation result perfomed by off-the-shelf OCR software.

II. Related Work

A. Background of Humane Studies

We may find some of the actual demands of the humanities researchers from the paper by Hayashi *et al.* [5], which includes the fact that the humanities researchers' actual demands sometimes differ from that of what information scientists expect. In the paper, it is said that 50% accuracy of a full text search will be deemed good for humanities researchers while it might be deemed bad for information scientists.

The paper [5] introduces the example of the literature study of handwritten documents. There are two methods to perform full text searches for handwritten documents: one is an OCR-based approach and the other is an image-based approach. In literature study, it is common that even the specialist cannot determine what is written in the image. In such a case, reading the "unreadable" word is in itself quite a professional activity. Assigning only one text to a word image, that is just the purpose of OCR, is not desirable, and it might even be harmful. On the other hand, an image-based approach is a safe solution for such documents. Even if image-based full text searching is not so fast and not so accurate, it still will be very useful for professional researchers.

Our research is the one intended not for handwritten documents but for historical typewritten documents. The paper [5] describes that image-based searching is also useful for historical typewritten documents. Since historical typewritten documents are "readable" material unlike in the case of handwritten documents, it seems that an OCR-based full text search is enough. However, OCR cannot afford high accuracy especially for historical Japanese documents because off-the-shelf OCR software is tuned for modern documents. If we need highly accurate OCR for historical documents, it will require specific tuning for it. However, specific tuning actually involves enormous cost. The advantage of an image-based approach is that it does not require specific tuning and is ready to apply to all languages in all ages.

III. Column Separation

To perform character segmentation with high accuracy, we first separate a whole newspaper image, which contains multi columns in a single page, into column-separated images.

A. Outer frame detection

In the layout of "Hakodate Shimbun", there is a thick outer frame surrounding a body text area. We detect it in order to confine the processing region.

Hough Transform is used for detecting outer frames. We used four subimages: upper half, lower half, left half and right half, to detect the outer frame's upper, lower, left and right edge, respectively. By this trick we could reduce the cost of line detection and improve the accuracy. Figure 3 represents an example of outer frame detection.

In Figure 3, black straight lines are detected by Hough Transform. A thick and upper straight line is estimated as the most suitable outer line. As the outer frame estimation, when two or more straight lines exist, the most outer line is assumed to be a most suitable outer frame.

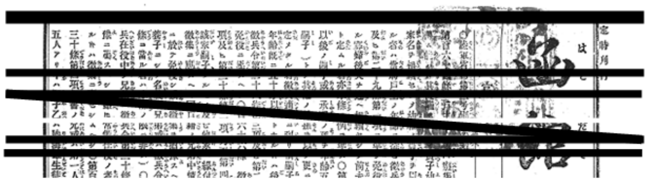


Figure 3. Result of outer frame detection. In several lines detected by standard Hough transform, the most upper and thickest line is estimated as the outer frame.

B. Column Separation

We separate an inner image of an outer frame into column-separated images by detecting ruled lines.

First of all, based on the layout of "Hakodate Shimbun," we detect transverse ruled lines that hold a major significance for the layout. To detect ruled lines, we used probabilistic Hough transform [6] instead of standard Hough transform. While standard Hough transform outputs lines without both endpoints, probabilistic Hough transform outputs line segments. Both endpoints of line segments can be useful for layout analysis.

Since probabilistic Hough transform is weak at line discontinuity, we perform dilation to the target image to remove line discontinuity. And since probabilistic Hough Transform sometimes outputs shorter lines than actual, we extend the detected lines until they reach to the outer frame. Figure 4 represents the result

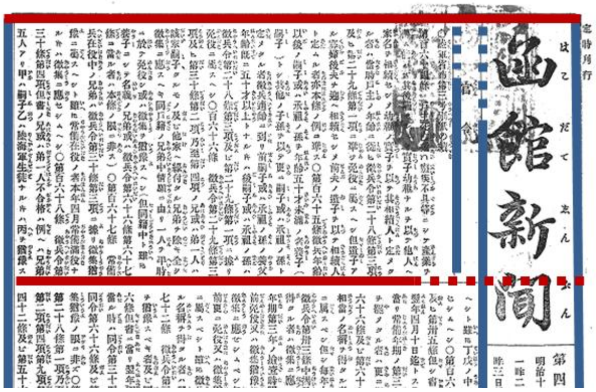


Figure 4. Result of ruled line detection and expansion of its result.

Simply applying the above method, we can obtain good results for the simple three-column layout, which form the majority of the dataset. However, to handle exceptional layout elements such as the masthead region, we added a trick to the process. Using knowledge of the target newspaper layout, if a vertical ruled line exists in the upper right area where a masthead is likely to exist, we assume that a masthead exists there.

IV. Character Segmentation

In this section, we propose the image processing technique to perform character segmentation on the column images with high accuracy.

A. Problems in segmentation using OCR software

To perform character segmentation, we first tried to use the off-the-shelf OCR software: "Typewritten document OCR library V7.0" (Media Drive Corporation). Figure 2 in Section I is an example of the software's output. From the figure, we can see that the result contains many wrong character segmentations. We carefully observed the result, and estimated that the following four factors mainly disturbed high-performance character segmentation.

- Ruled Lines
- Ruby characters
- Errors in Layout analysis
- Noise

Based on our observation, we decided to execute some preprocesses to remove these factors respectively. The proposed method removes ruled lines, ruby characters, and noise. The problem we suffered from the most is that the OCR software often finds one character over two or more lines (errors in layout analysis). To eliminate this disease, we decided to feed line-separated images as an input to OCR software, instead of the column-separated images.

B. Removal of ruled lines

In removing ruled lines, we applied the Run Length Smoothing Algorithm in [7]. In the original paper, connected components are extracted first, and then the rough width of text line is estimated using the size of connected components. Using the assumption that the size of the ruled line's connected component should be quite larger than the size of the connected component in the text, the larger connected components are regarded to be a ruled line and removed.

Also in the newspaper image, it is thought that the large majority of connected components represent single characters. Therefore, we presume that the width of the text line is the mean value of the width of all connected components. However, quite a lot of instances of minute size noise are actually included in the image, and the mean value of the width of connected components is affected from such noise. To overcome this problem, we ignore connected components whose size is less than ten pixels in the calculation. Thus we estimated the width of the text line. Connected components whose width is larger than the estimated text line width are regarded to be a ruled line and removed. The result of this process is shown in Figure 5.

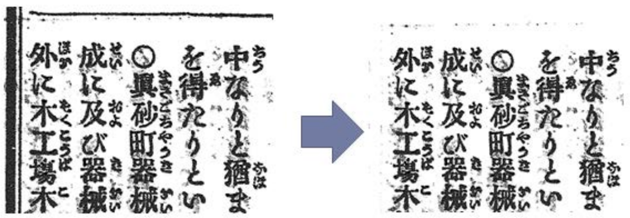


Figure 5. Removal of ruled lines

C. Removal of ruby characters

In removing ruby characters, we applied a scale space approach in [8], which was proposed for word segmentation of Latin handwritten document images. The difference between the original paper and our approach is that we tried to connect the character vertically.

First, the anisotropic Gaussian filter is applied to the target image. The anisotropic Gaussian function is defined as follow:

$$G(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left[-\left(\frac{x^2}{2\sigma_x{}^2} + \frac{y^2}{2\sigma_y{}^2}\right)\right] \qquad (1)$$

Using this anisotropic Gaussian filter, the object image is gradated vertically. Using the idea of scale space, the parameter σ_y is changed in steps, while the value of σ_x is fixed to 1. The gradated image is binarized by a certain threshold. We assumed that when the total area of connected components becomes the largest, the respective character regions are properly connected vertically. Figure 6 shows this process.

In Figure 6, we may observe that there are thin connected components that seem to be ruby characters on the right of thick connected components that seem to be a body characters. Among every connected region, we remove regions whose width is less than a certain threshold width. The threshold width is determined using the text line width estimated in the previous subsection.

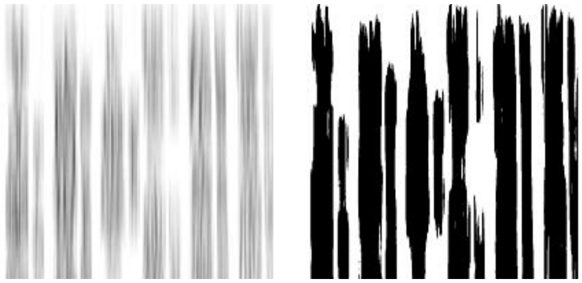


Figure 6. The left figure shows a result of applying a gaussian filter to a newspaper image, and the right figure shows the result of thresholding.

D. Text line segmentation

The OCR software we have used often made an error on the layout analysis for old documents. It often miss-recognized a region over two or more text lines as one character. To avoid this problem, we decided to feed line-separated images as an input to OCR software, instead of feeding column-separated images as an input to OCR.

To execute text line segmentation, the projection histogram that takes the sum of black pixels in the vertical direction is calculated, and the peaks which appear in the projection histogram are extracted. Minimal values of the histogram are assumed to be indicating the line borders.

E. Noise removal

Next, we try to remove noise caused from pollution or show-through. In order to detect such noise regions, the pixel value histogram is calculated for each line-separated image. Frequency in each pixel value from 0 to 255 is totaled in a histogram. Ideally, the histogram has two peaks: the majority background white regions (large pixel values) and the character black regions (small pixel values). In practice, however, regions where the intensities are lighter than character regions exist. In that case, the third peak appears in the histogram and that is regarded to be indicating noise regions.

Figure 7 shows the examples of the pixel value histogram. The histogram without instances of noise shows an ideal result. It has two peaks that indicate black and white, respectively. On the other hand, the histogram with instances of noise shows a different result. The third peak appears at the right of the black peak in the histogram. Then, this peak is assumed to be an instance of noise, and pixels whose value is close to the peak are removed. Figure 8 displays the result of applying this process.

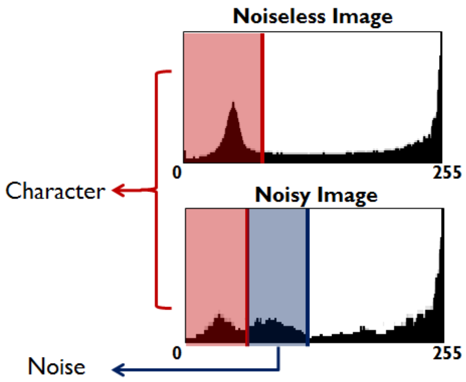


Figure 7. Histogram charts counting pixel values in a text line image.

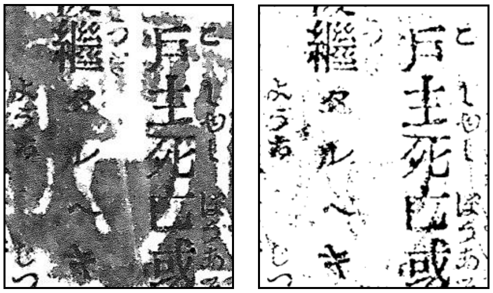


Figure 8. A text image that contains a lot of noise (left figure) and the result of removing noise from the left image (right figure)

F. *Result of the proposed method*

After all aforementioned processes are applied, the line-separated images are fed to the OCR software. Among the outputs, we discard the character recognition results (because their accuracy is far from satisfactory) and record the character segmentation results.

Figure 9 shows the affect of the proposed methods to character segmentation result. The left figure shows the result of character segmentation using only OCR software, and the right figure shows that of the proposed method. In the left figure, two or more characters are falsely recognized as to be one character. In the right figure, on the other hand, the character segmentation results are improved very much.

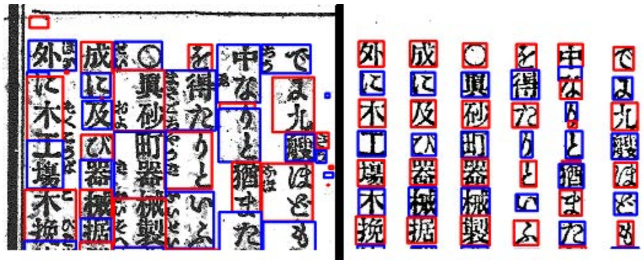


Figure 9. The left figure shows a result of character segmentation using only OCR software, and the right figure shows the improvement of character segmentation using the proposed methods.

V. Experimental Results and Discussion

In this section, we describe the performance evaluation of our method both in the viewpoint of column separation and character segmentation. Among 3,462 pages of our dataset, 771 pages are selected as the test set for the evaluation of column separation. Among them, 20 column images are selected for the evaluation of character segmentation. In each case the adequacy of the outputs is manually judged by visual confirmation.

A. *Experimental Results of Column Separation*

Table 1 shows the evaluation result of column separation. In this evaluation, the masthead region is counted as one column.

As displayed in the table, our method can correctly separate columns with 98.3% accuracy. The exceptions are 39 false-negatives and 61 false-positives.

All of the 39 false-negatives are in two-column layout pages (let me remind you that the majority of the dataset has a three-column layout). Such an irregular page layout causes false-negative detection.

All of the 61 false-positives are false detections of the masthead. This false detection occurs in the place where two ruled lines accidentally exist at the same place as the masthead.

TABLE I. Result of Column Separation

Number of Columns	Detected Columns	Correctly Detected Columns	Accuracy	False-negatives	False-Positives
2379	2401	2340	0.983	39	61

B. *Experimental Results and Discussion of Character Segmentation*

Table 2 shows the evaluation result of character segmentation. The upper row displays the result when off-the-shelf OCR is used without any preprocessing. The lower row displays the result of the proposed method.

TABLE II. Result of Character Segmentation

	Number of characters	Detected regions	Correctly detected regions	Accuracy
Without preprocessing	17021	10737	5375	0.316
Proposed method	17021	17223	16384	0.963

TABLE III. CAUSE STATISTICS OF FALSE DETECTION RESULT OF COLUMN SEPARATION

	Over-segmentation	Connecting multiple characters	False-negative detection	Ruby characters
Without preprocessing	428	9913	102	2197
Proposed method	626	40	10	191

As seen in Table 2, the proposed method achieved significantly high accuracy compared with the "without-preprocessing" condition. To observe the result more in detail, we have made cause-specific statistics of the false detection. The result is summarized in Table 3. In the table, four typical causes of false detection are displayed. 1) Over segmentation is the case when the system outputs multiple segments for single characters. 2) Connecting multiple characters is the case when the system outputs a single region for multiple characters. 3) False-negative detection is the case when the system cannot detect the character even though it exists there. 4) Ruby character is the case when the system detects characters even when they are not supposed to be there.

From the table, we may observe that the biggest advantage of the proposed method is that it can decrease the multiple-characters-connection. We presume this advantage comes from the fact that we decided to use line-separated images instead of column-separated images.

We are afraid that ruby character detection errors still remain (even though they are greatly decreased). However, we may say that it does not matter in practice because we may easily remove ruby characters by postprocessing using a difference between the size of the body characters and the size of the ruby characters. The purpose of the ruby removal in preprocessing is to avoid a wrong connection of ruby characters and body characters together, and the purpose is properly achieved.

We are also afraid that the number of over-segmentation instances is increased when our method is applied. It is thought that the proposed noise removal technique causes over-segmentation. Because several character strokes become interrupted by the noise removal technique, characters are excessively divided. Figure 10 displays an example of excessive division.

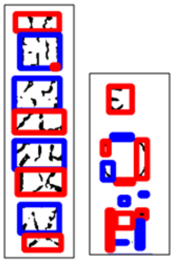


Figure 10. Examples of over-segmentation

VI. CONCLUSIONS

We propose an image processing technique to perform highly accurate character segmentation in order to realize a full text searching function for the Hakodate Shimbun, a historical Japanese newspaper published in the 1880s. While the accuracy of the existing character segmentation technique was less than 35%, the proposed method improves it more than 95%. By combining the proposed method and our previous study, it becomes possible to construct a highly accurate full text searching system to large-scale document image archives. The complete system is already implemented and publicly in use via the Hakodate City Central Library website [9].

A. Future research

In the proposed method, tables and advertisements are excluded from the objectives. In our future research, we are going to try to include them to the objectives and appropriately process them. The suppression of over-segmentation is also included in our future works.

REFERENCES

[1] Database, National Institute of Japanese Literature, Japan, http://www.nijl.ac.jp/pages/database/

[2] Digital Library from the Meiji Era, National Diet Library, Japan, http://kindai.ndl.go.jp/

[3] K. Terasawa, and Y. Tanaka, "Slit Style HOG Feature for Document Image Word Spotting," Proc. Int. Conf. on Document Analysis and Recognition, ICDAR2009, pp. 276-280, 2009.

[4] K. Terasawa, T. Nagasaki, and T. Kawashima, "Eigenspace method for text retrieval in historical document images," Proc. Int. Conf. on Document Analysis and Recognition, ICDAR2005, vol. 1, pp. 437-441, 2005.

[5] S. Hayashi, K. Nagai, and I. Miyazaki, "Information Technologies for Humanities – A View of Researchers of History and Classics – ," Journal of the Japanese Society for Artificial Intelligence, vol.25, no. 1, pp.24-31, Jan. 2010.

[6] N. Kiryati, Y. Eldar, and A. M. Bruckstein, "A probabilistic Hough transform," Pattern Recognition, Vol. 24, Issue 4, pp.303-316, 1991.

[7] N. Stamatopulos, G. Louloudis and Basilis Gatos, "A Comprehensive Evaluation Methodology for Noisy Historical Document Recognition Techniques," Proceedings of the Third Workshop on Analytics for Noisy Unstructured Text Data, AND2009, pp.47-54, July 2009.

[8] R. Manmatha, C. Han and E.M. Riseman, "Word spotting: A new approach to indexing handwriting", Proc. IEEE Conf. on Computer Vision and Pattern Recognition, CVPR'96, pp. 631-637, June 1996.

[9] Hakodate Central Library, "Document Image Full Text Searching System," http://www.lib-hkd.jp/rein/