人文科学とコンピュータ 55－８
（２００２．７．２６）

古文書ＯＣＲのための文字切り出し

原　正一郎
国文学研究資料館研究情報部

本稿では、古文書 OCR の前処理として必須である、つづき文字を切り出す手法について提案する。提案する方法では、まず簡単なフィルタ処理（色に基づき文字の候補となるピクセルを抽出するカラーフィルタ、ゴマ塩雑音などの除去フィルタ、カラー画像を白黒階調さらに２値化するフィルタ）により、雑音の少ない良好な２値画像を作成する。次に周辺分布からページごとの平均文字サイズ、縦書き・横書きに関するレイアウト情報を抽出する。文字はこれらの情報に基づいてピクセルから組み立てる。つまり隣接するピクセルを集めて文字のセグメントを生成し、次いで近傍のセグメントを集めて文字あるいはつづき文字を生成する。つづき文字の切り出しは文字輪郭上の相対する凹部分を結ぶ線に沿って行う。本法の特徴は、適切な凹部分を画像の多重解像度解析に基づいて発見するところにある。

Segmentation of Cursive Characters for Classical Literal OCR

Shoichiro Hara
National Institute of Japanese Literature,
Address: 1-16-10 Yutaka-cho, Shinagawa-ku, Tokyo 142-8585 Japan,
Fax: 81-3-3784-8875
E-mail: hara@nijl.ac.jp

This paper is a proposal for new means of separating a cursive character into separate and distinctive characters for further OCR processing. The proposed method begins with some filtering, i.e., a color filter to extract candidate pixels of characters according to their color, a noise reduction filter, a conversion of a color image to a gray image, and a binarization. Layout information as to whether a text is written vertically or horizontally as well as average character size (ACS) in a page is obtained from the analysis of a peripherally projected histogram. A character is constructed gradually from pixels. First, connected pixels are aggregated to a small segment. Then neighboring segments are collected to a character or a cursive string. At last, segmentation of a cursive string is basically made along the line connecting the concavity on a contour and its vicinity concavity on the opposite contour. The strength of the new method is to find appropriate concavity by multiresolution analysis.

1. INTRODUCTION

The bottleneck of electronic transcription of classical materials is data input, and OCR (Optical Character Recognition/Reading) technique is expected to accelerate this process. The latest OCR technique shows high performance on modern printed materials. However, its availability to classical materials is extremely restricted. Thus, the advanced OCR for handwritten characters in classical materials is crucial for researchers who are engaged in digitizing classical materials.

The primary difficulty of handwritten character recognition comes from shape variations due to writers' habits, styles, and times, then finding appropriate features and fast algorithms have been the main focus in this research. There are many papers on this theme, but most of the studies have used controlled characters, that is, characters are extracted manually from original images and carefully organized. Some studies treated cursive scripts, but the targets were restricted to numbers and alphabets [1,2,3,4]. However, from the practical point of view, the essential difficulty is the extraction of characters from the

original images. Classical papers often suffer from wormholes and discoloration due to aging, and there are sometimes seals and annotations overlapped on characters. These make character extraction difficult. Moreover, characters in classical texts are often cursive. Thus, the segmentation of a cursive string into some characters is important. This process is categorized as preprocess in OCR. Unfortunately, there are few papers concerning the details of issues.

This paper proposes a series of preprocesses for OCR. The proposed method begins with some filtering, i.e., a color filtering filter to extract candidate pixels of characters according to their color, some noise reduction filters, a conversion of a color image to a gray image, and a binarization. Layout information as to whether a text is written vertically or horizontally as well as average character size (ACS) in a page is obtained from the analysis of a peripherally projected histogram. A character is constructed gradually from pixels. First, connected pixels are aggregated to a small segment. Then neighboring segments are collected to a character or a cursive string (**segment aggregation**). At last, segmentation of a cursive string is basically made along the line connecting the concavity on a contour and its vicinity concavity on the opposite contour (**character segmentation**). The strength of the new method is to find appropriate concavity by multipresolution analysis.

2. PREMISES ON MATERIALS

The characters in classical materials have various forms. As a preliminary study, some restrictions are imposed on materials as follows:

[Premises on Text Pages]

1) A text page looks like a homogeneous texture over an image.
2) A texture comprises foreground color blobs that correspond to characters and a background color plain that corresponds to paper (Fig.1).
3) Blobs stand in horizontal or vertical lines, but both types do not appear simultaneously.
4) Strings are aligned regularly.
5) Foreground color is black and background color is gray.

[Premises on Characters]

1) Character size in an image is almost the same.
2) Characters are mostly separated by background color.
3) A character is constructed from small segments and each of segments can be enveloped by a rectangle.
4) A character can be enveloped by a square (Fig.2).

As a whole, example images do not include pictures nor ruled lines. Characters are mostly written by the square style, that is, some characters may adjoin each other but most characters are separated. These premises are a little bit strict. But even under these restrictions, there are many materials that satisfy these premises.

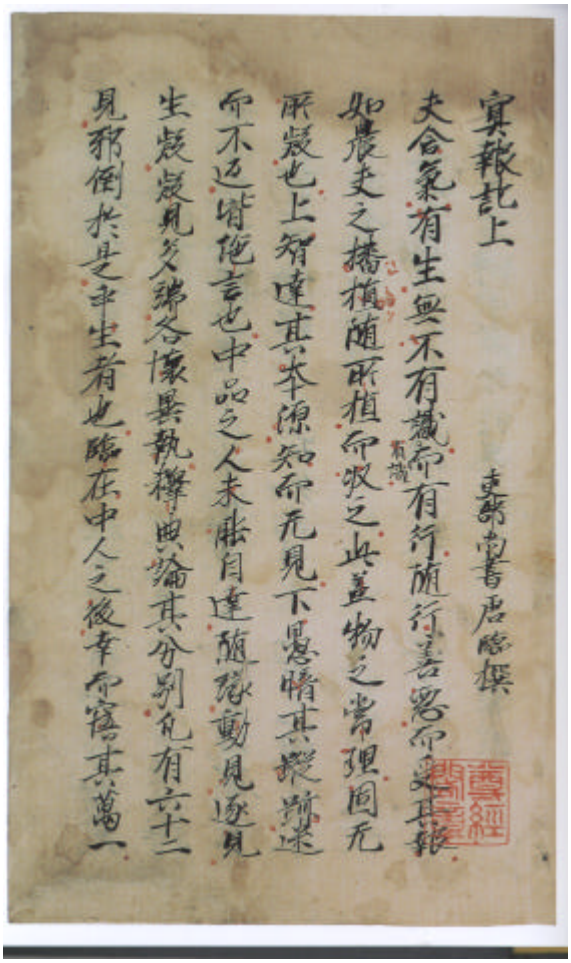

Fig.1 Sample Text Image

Fig. 2 Character

3. PREPROCESSES

Some preprocesses are used to create segments of characters. Following are the lists of preprocesses. Each

process is applied in this order. Among these, former four processes are used to make a fine binary image from an original color image. The rest of processes are used to construct elementary segments and to extract useful information for the following aggregation and segmentation processes.

1) Color Filtering: This filter removes non-gray pixels from an original image. The rest of pixels will construct characters.
2) Color Interpolation: This filter interpolates the color of pixels that are removed by "Color Filtering."
3) Color Conversion: This filter converts an RGB image to a gray level image.
4) Binarizing: This filter binarizes a gray level image. The threshold is calculated based on the Otsu's Discrimination Method [5].
5) Global Filter: A horizontal histogram counting each row of black pixels and, in similar way, a vertical histogram are built. Then Fast Fourier Transform is applied to these histograms to estimate average character size (following ACS) of a page image and to estimate the direction of text lines.
6) Connective Analysis: This process collects adjacent pixels to construct a continuous segment that might correspond to a part of a character. The connective analysis [6] is used to perform this process. Some experiments showed that 4-connection gives the smoother contour of a segment than 8-connection.
7) Small Segments Elimination: This filter eliminates small segments that might be noises.
8) Feature Extraction: While aggregating small segments to a character, some feature measurements of each segment can be used.
9) Strange Segments Elimination: This filter eliminates non-character segments such as lines, dots etc.

4. AGGREGATION OF SEGMENTS

Followings are simple explanations of the segment aggregation algorithm.

[Segment Expression]
Each segment is enveloped by a rectangle that is denoted by two pairs of coordinates.

[Definitions of Distance]
A kind of distance is defined to measure the nearness of segments.

[Segment Aggregation Algorism]
The basic idea of the segment aggregation is to select the smallest segment, to find its nearest segment in an image, to merge them, and to create a new segment. This process is repeated until a certain condition will be satisfied.

5. SEGMENTATION OF CURSIVE CHARACTERS

In the preceding aggregation procedure, most of the segments are grouped up to correct characters, but there are several miss-grouped segments. Most of these segments belong to cursive characters. This section describes the method to segment cursive characters. The basic idea is to find appropriate lines that separate a cursive character into characters. From the premises, these separation lines lie at each ACS. The key to draw separation lines is that each line passes a pair of concavities.

[Concavity]
Suppose a smooth continuous function f(x,y), a concavity can be found by calculating the second derivative f"(x,y). That is, if f"(x,y)≧0, then gradient of the tangent line increases, and this means a concavity [7]. This study uses this intuitive method to find concavities. To find concavities, the contour tracing method is used. The tracing is done clockwise looking at the character shape to right, and produces the differences of the angles between neighbor tangent lines. These differences are the same as the second derivatives of the contour, and its local maximum means the local bottom of a concavity.

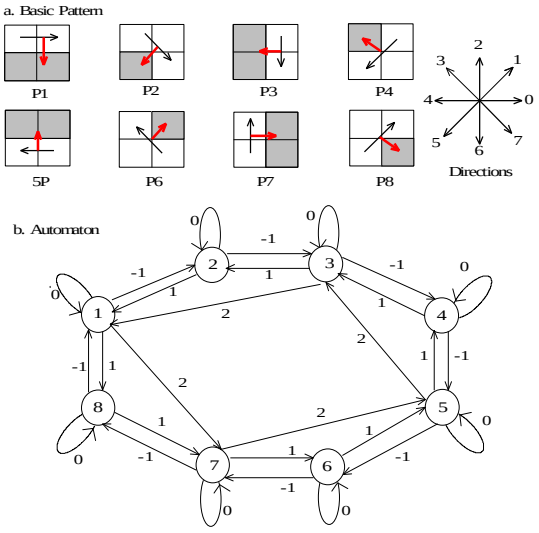

Fig.3 Contour Tracing

[Contour Tracing]
The simple probe is introduced to trace contours. This probe moves on white pixels along black pixels and traces a contour clockwise looking at the character pixels to right (Fig. 3). The probe will find one of 8 patterns in each position while tracing as shown in Fig. 1-a. Each pattern can be expressed as 2 by 2-pixel matrix. A black pixel is part of a character and a white

pixel is part of the background. A thick arrow is a tangent line that indicates a tracing direction, and its starting pixel is the position of the probe. A thin arrow shows the look of a probe. As there are 8 contour patterns, 8 tangent directions are defined and denoted as 0 to 7 as in the Fig.3-a. The probe begins its linear scanning from the top-left pixel of an image until finding a black pixel. If the black pixel is found and it is not marked, contour tracing begins. While the contour is being traced, the probe puts marks along the contour to avoid repeating the same contour tracing. This scanning and tracing procedure is repeated until there are no remaining contours to be examined.

During one contour tracing, the probe generates the difference of the angles between neighbor tangent lines according to the contour pattern and the automaton shown in Fig.3-b. In the figure, numbers in circles are the patterns, and arrows indicate the transitions among patterns, and numbers attached to arrows are the outputs from the automaton. These output numbers correspond to the differences of angles between neighbor tangent lines. The 1 unit is equal to $\pi/4$ of angle difference. For example, the next possible patterns from the pattern 1 are the pattern 1, 2, 7, or 8. If the pattern 2 is followed by the pattern 1, the automaton generates "-1," which means the tangent angle decreases $\pi/4$. One contour tracing is finished when the probe finds the initial black pixel it begins tracing.

[Concavities Extraction]

Concavities are defined by the local maxima that are greater than the threshold value. The threshold is determined by ACS and standard deviation generated from the automaton in Fig.3. Several concavities are sometimes close to one another within a predetermined distance. In this case, only the earliest concavity is used as a representative of this specific area.

[Separation Lines]

Separation lines are defined by the following three procedures that are an expansion of a method devised by Holt et al. [2]. The first procedure is to generate all candidates of separation lines.

Generation Rules

1) A separation line passes through the nearest two concavities.
2) A separation line traverses a character region only once, that is, a separation line that passes white pixels is eliminated (line (a) in Fig.4).
3) One concavity has at most one separation line. If a concavity has more than two lines, the shortest line remains and others are eliminated.
4) A separation line is not long, that is, a line longer than the predetermined length is eliminated.

The generation rules generate many lines that are not relevant to the correct separation lines. Following elimination rules are used to remove unnecessary lines.

Elimination Rules

6) If a contour has only one contour where a separation line starts or ends, this line is eliminated. *Line (b) in Fig.4 shows the example. The contour L has only one line that starts from the contour "L" and ends at the contour "I."*
7) If a text is horizontal and the height of the fraction cut off by a separation line is shorter than predetermined length, this line is eliminated. *The line (c) in Fig.4 shows the example. The fragment created by the line (c) is very small. This means that line (c) does not separate a character but separates parts of a character*.
8) If there are two separation lines that are very close to each other and that act as a link between same contours, the longer line is eliminated. *Lines (d) and (e) in Fig.4 show the example. As these two lines are very close to each other, one of them (in this case, line (d)) should not be a separation line*

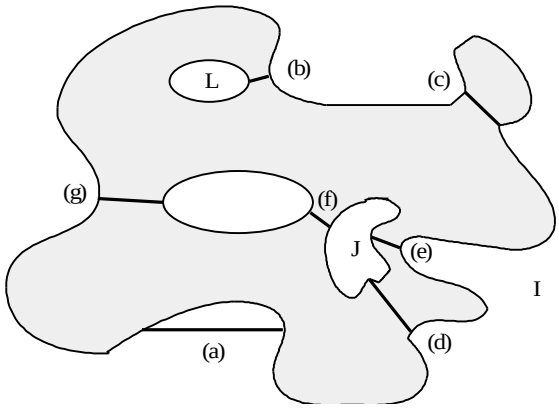

Fig4. Separation Lines

Final Decision Rules

Then the final decision rules are applied to extract separation lines.

9) A separation line lies at ACS intervals.
10) A separation line must start from and end at the same contour. However, if a line does not end at the starting contour, there must be more lines drawn to connect the starting contour. *Lines (e), (f), and (g) in Fig.4 show the example*

6. APPLYING MULTIRESOLUTION ANALYSIS TO WINNOW SEPARATION LINES

From the premises on characters as described in section 2, characters' sizes in a page must be almost same. Thus ACS is a strong parameter to select separate lines.

The Rule 9 in "Final Decision Rules" is depend on this premises. However, as most of classical materials are hand-written texts, not all characters in a page are constant size. Some characters have much different sizes from ACS, in which case, ACS alone would select wrong separation lines.

To compensate for this defect, a kind of multiresolution analysis method is introduced [8]. The basic idea is that an original image I(x,y) is blurred by convoluting a Gaussian function G(σ) such as G*I, then Laplacian operator ∇^2 is applied such as ∇^2(G*I) = (∇^2G)*I. Most of edges of characters are expected to be detected by (∇^2G)*I = 0. This is called "zero-crossing." The Gaussian function behaves as a band pass filter that wipes out small structures at scale less than the parameter σ (standard deviation).

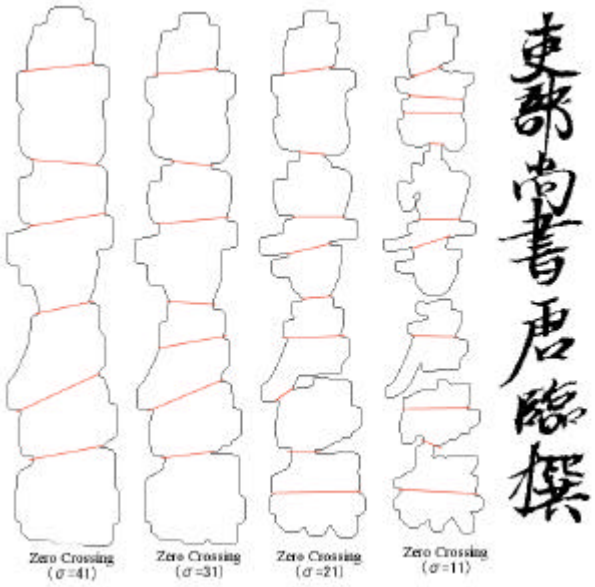

Fig.4 Segmentation Lines using ∇^2G

Figure 4 shows the effect of ∇^2G. The right picture is an original binary image. The rest of pictures are obtained by convoluting ∇^2G to the original binary image with different σ and detecting outermost zero-crossings. Lines traversing zero-crossings are candidates of separation lines obtained by the same rules as section 5.

These pictures show how an image looks like with σ. The picture with smaller σ conserves detail shape. When σ becomes larger, a picture becomes rougher. In the physical world, if a zero-crossing line is present in a contiguous range of frequency channel and the line has the same position in each channel, this indicates the presence of the intensity change in an image [9]. In the same sense, a concavity conserved in a rougher picture means that shape changed largely around there. The important issue is that the large change of shape in a rougher picture is also conserved in the detail picture, that is, separation lines in a rougher picture must exist in the detail picture. The location of a separation line is different with σ, but the location can be obtained correctly by tracing the separation line of smaller σ.

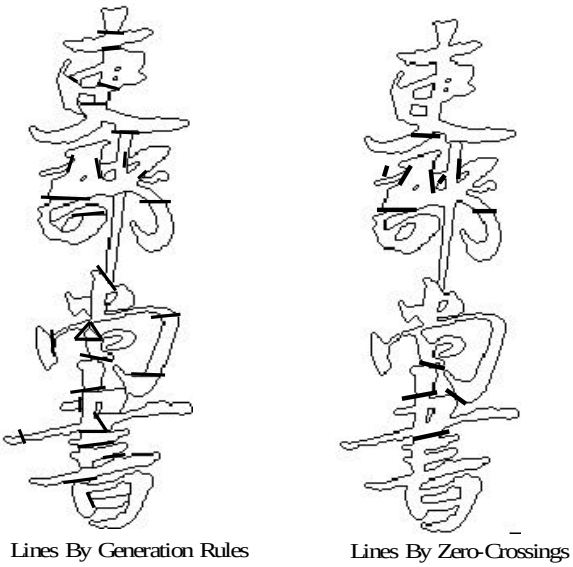

Fig.5 Winnowing Separation Lines

Fig.5 shows how the number of candidates of separating lines decreases by multiresolution analysis method. The left picture shows the candidates of separation lines by "Generation Rules" in Section 5. The candidates of separation lines in the right picture in Fig.5 are compared with lines of picture with σ=11 in Fig.4. If the locations of two lines from both pictures are near to each other, the lines from the left picture in Fig.5 remains as a relevant line. The right picture is the shows after winnowing irrelevant lines.

7. CONSIDERATION AND CONCLUSIONS

The new means of separating a cursive character into characters is proposed. This proposed method begins with some filters, i.e., a color filtering, a noise reduction, conversion of a color image to a gray image, and binarization. Layout information as to whether the text is written vertically or horizontally, as well as the average character size of the text, are obtained from an analysis of a peripherally projected histogram. A character is constructed gradually from pixels. First, connected pixels are aggregated to a small segment by connection analysis. Then neighboring segments are aggregated to a character or a cursive string. Several rules and an evaluation function are introduced for aggregation. Next, the segmentation procedure is applied to each cursive character in order to separate them into characters. Separation is basically made along the line connecting a

concavity on a contour and its nearest concavity on the opposite contour.

The strength of this proposed approach avoids the need for language specific character style knowledge and layout information. Though under the strict initial conditions introduced in the section 2, the proposed methods can correctly extract characters from images.

However, as most of classical materials are hand-written texts, not all characters in a page are constant size, which leads wrong segmentations. To compensate for this defect, a kind of multiresolution analysis method is introduced. By the small sample examinations, this method shows the possibility for effective elimination of irrelevant separation lines.

The ability of $\nabla^2 G$ is depend on the appropriate selection of σ. In the case of Fig.1, ACS is about 100 (pixels) , and Fig.4 shows σ=30 is appropriate. This is about 1/3 of the ACS, that is, Gaussian distribution with σ=30 covers almost width of 100 pixels. On the other hand, wavelet transform (Harr wavelet) is applied to the same binary images and calculated the powers of each resolution. Table 1 is the example. This shows that level –4 and –5 (pixel sizes is about 30) include much information and this value is almost same value obtained by ACS.

Level	Pixel Size	Power
-2	4 by 4	25058
-3	8 by 8	31690
-4	16 by 16	44374
-5	32 by 32	38376
-6	64 by 64	24682
-7	128 by 128	9844
-8	256 by 256	4113
-9	512 by 512	10619
-10	1024 by 1024	2400

Table 1 Example of Power and Pixels Size by Harr Wavelet Transform

At present, only a few samples have been examined, thus it is too early to precisely evaluate this series of procedure.

REFERENCES

[1] R. G. Casey and H. Takahashi, “Experience in segmenting and classifying the NIST data base,” From Pixels to Features III: Frontiers in Handwriting Recognition, Elsevier Science Publishers, pp. 5 - 16, 1992.

[2] M. J. J. Holt, M. Mohammad Beglou and S. Datta, “Slant-independent letter segmentation for off-line cursive script recognition,” From Pixels to Features III: Frontiers in Handwriting Recognition, Elsevier Science Publishers, pp.41 - 46, 1992.

[3] R. Fenrich, “Segmentation of automatically located handwritten numeric strings,” From Pixels to Features III: Frontiers in Handwriting Recognition, Elsevier Science Publishers, pp. 47 - 59, 1992.

[4] H. Nishida and S. Mori, “A Model-Based Split-and-Merge Method for Character String Recognition,” Document Image Analysis, World Scientific, pp.1205-1222, 1994.

[5] N. Otsu, “An automatic threshold selection method based on discriminant and least square criteria,” Trans. IECE Japan, Vol. J36-D, No.4, pp.349-356, 1980.

[6] H. Bassmann and P. W. Besslich, “Ad Oculos, Digital Image Processing,” Student Edition 2.0, Thomson Publishing, 1995.

[7] S. Mori, H. Nishida and H. Yamada, “Optical Character Recognition”, John Wiley & Sons, Inc., 1999.

[8] C. K. Chui, “An Introduction to Wavelets,” Academic Press, 1992.

[9] D. Marr, “VISION,” W.H.Freeman and Company, 1982.