Isolated Character Recognition by Searching Feature Points

Masakazu IWAMURA,* Kazuya NEGISHI,† Shinichiro OMACHI,† and Hirotomo ASO†
*Osaka Prefecture University, 1–1 Gakuencho, Sakai-shi, Osaka, 599–8531 Japan
†Tohoku University, 6–6–05 Aoba, Aramaki, Aoba-ku, Sendai-shi, 980–8579 Japan

Abstract

Conventional segmentation technique cannot extract difficult characters such as an isolated character and a touching character. In this paper, we propose a novel character recognition method which executes segmentation and recognition simultaneously. This method enables us to extract and recognize such difficult characters. The effectiveness of the proposed method is confirmed by experiments.

1. Introduction

Accuracy of character recognition for segmented character image is enough for practical use. However, conventional segmentation method can be applied only *well-defined* problems such that the characters constitute a line, completely separated from other characters and so on. Therefore, an *isolated character* which does not constitute a line of characters, and a touching character which connects to other characters are hard to extract. In this paper, we propose a novel character recognition method which allows segmentation of an isolated character, a touching character and so on by executing segmentation and recognition of a character simultaneously.

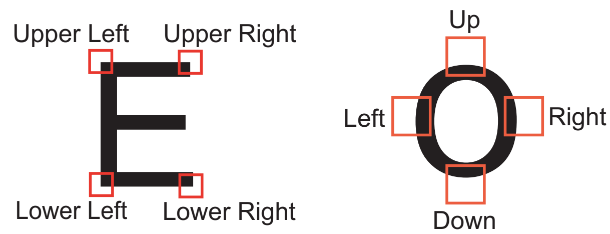
2. Preparation

2.1. Input image and reference image

An *input image* is a relatively larger image such as a photograph including isolated characters. A *reference image* is a relatively smaller image among the 52 alphabet letters and 10 numerals.

2.2. Histogram of edge directions

At first, the edge direction and the edge intensity of each pixel is calculated with the Sobel filter. Then, the edge direction is quantized into n, and a histogram whose bin width is $\theta = \frac{2\pi}{n}$ radian is constructed. To eliminate the influence of noise, a pixel whose edge intensity is less than a threshold is regarded as a *no edge pixel*, and the histogram does not contain the *no edge pixels*. The histogram is normalized so that the sum of all the bins values to be 1. This normalized histogram is used as a feature vector of n dimensions.



(a) Corners. (b) Curves.

Figure 1. Features.

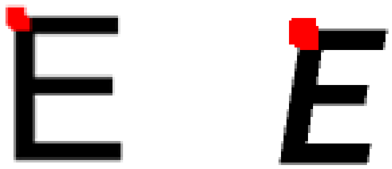
2.3. Features

A corner, a curve, a branch, an intersection and a bending point has been considered as effective features in character recognition. In the proposed method, four corners and four curves are used as features because they are easy to extract by search and seem to be effective. Four corners are "upper left", "upper right", "lower left" and "lower right" (see Fig. 1(a)). Four curves are "up", "down", "left" and "right" (see Fig. 1(b)).

To extract features robustly, larger θ is desired. However, to extract complex features, smaller θ is required. Therefore, θ depends on features: $\theta = \pi/2$ (i.e., $n = 4$) were used for corners, and $\theta = \pi/8$ (i.e., $n = 16$) were for curves.

Four intervals of bins of a corner are defined as

$$\left[-\frac{\pi}{4} + \frac{\pi}{2}n, \frac{\pi}{4} + \frac{\pi}{2}n\right), \quad 0 \leq n \leq 3. \tag{1}$$



(a) Arial. (b) Franklin.

Figure 2. Robust feature point extraction of the upper left feature.

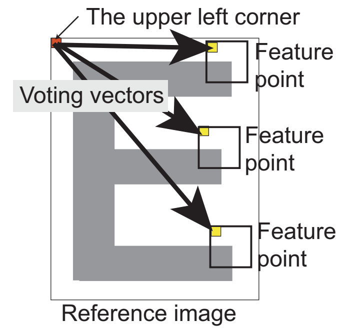


Figure 3. Feature point extraction from the reference image. Three feature points are extracted as the upper right corner.

Such a loose feature can extract both regular font such as Arial and oblique font such as Franklin as shown in Fig. 2. Eight intervals of bins of a curve are defined as

$$\left[-\frac{\pi}{16}+\frac{\pi}{8}n, \frac{\pi}{16}+\frac{\pi}{8}n\right), \quad 0 \leq n \leq 15. \tag{2}$$

2.4. Similarity

The proposed method searches for the eight kinds of features from all rectangular area of 5×5 pixels in the input image and the reference images. The rectangular area is called the *searching area*.

To evaluate how much the searching area is similar to each feature, a similarity measure proposed by Swain et al. [1] is used. Let H be the histogram of the searching area, M be the histogram of a corner feature or a curve feature. Then, the similarity between H and M are

$$S_{HM} = \sum_{t=1}^{Q} \min(H_t, M_t), \tag{3}$$

where H_t and M_t are the t-th bin value of H and M respectively, and Q is the number of histogram bins of the feature (i.e., $Q = 4$ or $Q = 16$). Since both H and M are normalized histogram, $0 \leq S_{HM} \leq 1$. If the similarity of the searching area is more than threshold T_1, $0 \leq T_1 \leq 1$, the searching area is regarded as a feature point.

3. Segmentation and recognition of isolated characters

3.1. Registration of reference image

First of all, all the feature points are detected from the reference images, and the positions of them are registered.

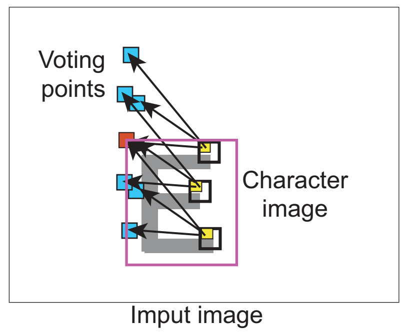


Figure 4. Feature point extraction and segmentation from the input image. There are three voting points per feature point.

The information of the position is represented by *voting vectors* (see Fig. 3): Let the origin be the upper left corner of the reference image. Let $N_i^{(k)}$ be the number of occurrences of i-th feature in the reference image of character k. Let (x_{ij}, y_{ij}) be the coordinate of j-th occurrence of i-th feature in the character k. The voting vector is defined as

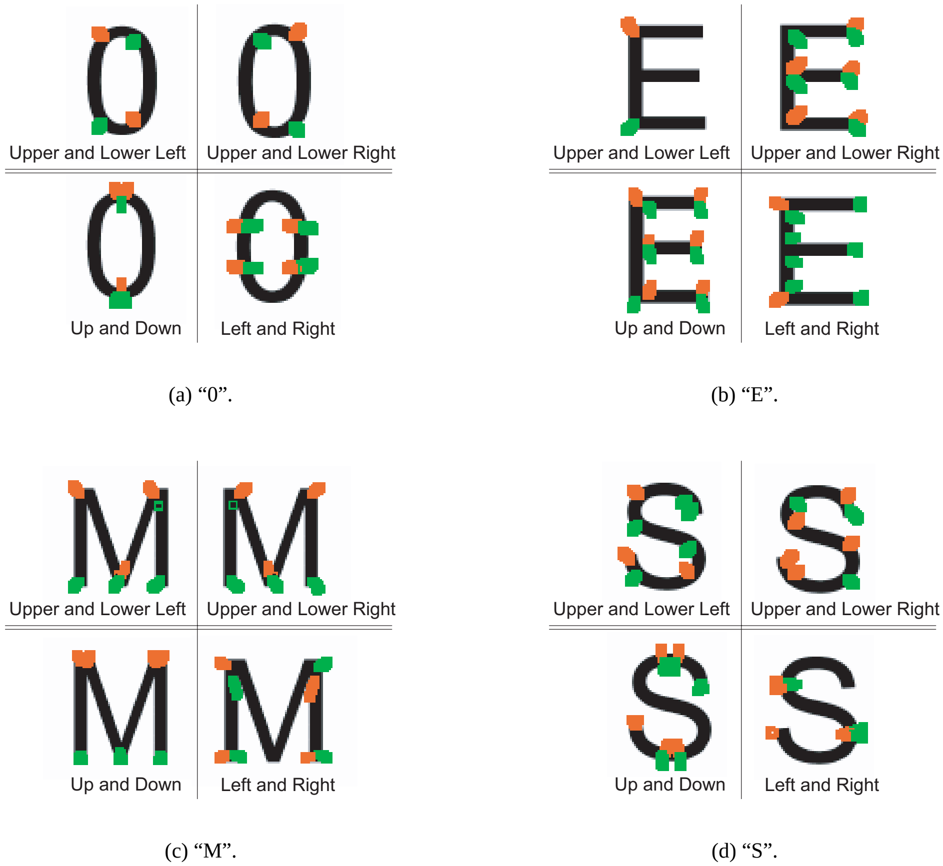
$$\boldsymbol{f}_{ij}^{(k)} = (x_{ij}, y_{ij}), \quad 1 \leq i \leq 8,\ 1 \leq j \leq N_i^{(k)}. \tag{4}$$

Furthermore, the total number of occurrences of eight features in the reference image of character k is defined as

$$F_S^{(k)} = \sum_{i=1}^{8} N_i^{(k)}. \tag{5}$$

3.2. Segmentation

At first, all the feature points are detected from the input image. Then, characters are detected by voting. At the



(a) “0”. (b) “E”. (c) “M”. (d) “S”.

Figure 5. Feature points. Corners are in the upper row, and curves are in the lower row.

same time, category of the character and its location are determined by voting at the *voting point* $\boldsymbol{P}_{ijl}^{(k)}$ given as

$$\boldsymbol{P}_{ijl}^{(k)} = \boldsymbol{x}_{il} - \boldsymbol{f}_{ij}^{(k)}, \quad 1 \leq j \leq N_i^{(k)}, \text{ for all } k, \qquad (6)$$

where $\boldsymbol{x}_{il}$ is the l-th occurrence of the i-th feature in the input image. In short, the voting point is a candidate of the upper left corner of a character image (see Fig. 4).

To allow deformations of a character image, the neighbors of the voting point are also voted. The voting rule is as follows: Let $v_{ij}^{(k)}(x, y)$ be the *voting value*, which stands for the voting value of character k at (x, y) of the input image. In advance, it is initialized as $v_{ij}^{(k)}(x, y) = 0$ for all (x, y) and all k. The neighbors within r pixels distance from the voting point are voted as

$$v_{ij}^{(k)}(x, y) = 1, \quad \text{for } (x, y) \text{ s.t. } \left|(x, y) - \boldsymbol{P}_{ijl}^{(k)}\right| \leq r. \qquad (7)$$

In this paper, $r = 8$ was used. Note that even if the same point is voted more than once, the voting value is 1.

After the voting, let $V^{(k)}(x, y)$ be the sum of the voting values $v_{ij}^{(k)}(x, y)$. Namely,

$$V^{(k)}(x, y) = \sum_i \sum_j v_{ij}^{(k)}(x, y). \qquad (8)$$

$V^{(k)}(x, y)$ stands for the possibility that the upper left corner of the character image of category k exists at (x, y). Let T_2 be the threshold of character segmentation where $0 \leq T_2 \leq 1$. If

$$V^{(k)}(x, y) \geq F_S^{(k)} T_2 \qquad (9)$$

is satisfied, (x, y) is determined to be the upper left corner of an image of character k. Here, $F_S^{(k)} = \max V^{(k)}(x, y)$.

4. Experiments

4.1. Feature point extraction

Feature points extracted by the proposed method are illustrated in Fig. 5. The figures are feature points of the ref-

Figure 6. Correct: the reference image is “A”.

erence images of “0”, “E”, “M” and “S” (Arial). In the figures, each rectangle represents a feature point. Because feature points detected were so crowded, rectangles look filled. 5×5 was used as the size of the searching area. $T_1 = 0.75$.

In the figures, other features than pure corners and curves were extracted. However, the most important thing to recognize correctly is that the same points are extracted from the reference image and the input image. Therefore, they were used without eliminating.

4.2. Recognition experiments

To confirm the effectiveness of the proposed method, three experiments were carried out against (1)different fonts, (2)isolated characters, and (3)touching characters. Preliminary to all the experiments, smoothing filter was applied to all the reference images and the input images.

The proposed method ignores the *no edge pixels*, and does not use the information of them for segmentation and recognition. However, combination of many *no edge pixels* and a few edge pixels in a searching area can cause miss detection of the feature points. Therefore, the searching area which contains more than 70% of *no edge pixels* is ignored.

Experiment results are classified into three categories for evaluation: “Correct”, “Match”, and “Miss”. If all the character images of the same category as the reference one are extracted, it is classified into “Correct” or “Match”. If the extracted images consist of only the same category as the reference one, it is classified into “Correct” (see Fig. 6). If not, it is classified into “Match” (see Fig. 7). If all the character images of the same category as the reference one are not extracted, it is classified into “Miss” (see Fig. 8).

4.2.1 Against different fonts

The input images of several fonts, similar to Figs. 6 to 8, were prepared. When the fonts of the reference image and the input image are the same, recognition rate was 100%. Therefore, experiment results against different fonts

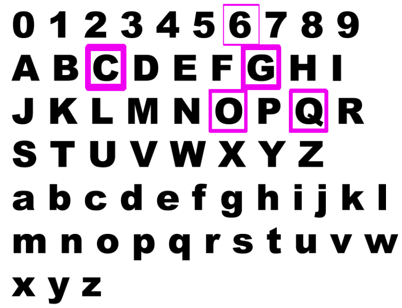


Figure 7. Match: the reference image is “C”.

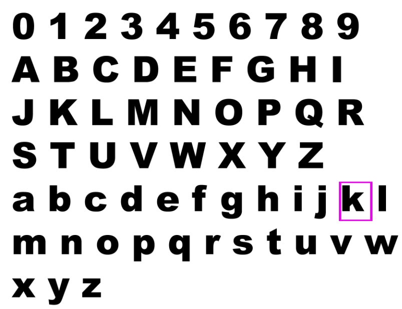


Figure 8. Miss: the reference image is “K”.

are shown here. Arial was used for the reference images and Arial Black, Franklin and Maru Gothic were used for the input image. Arial Black is bold, Franklin is oblique, Maru Gothic has rounded corners. Two thresholds were determined as $T_1 = 0.75$ and $T_2 = 0.85$.

Recognition results are summarized in Table 1. The table shows that almost 30% were classified into “Correct”. Also, more than 95% of Arial Black and Franklin, and more than 70% of Maru Gothic were classified into either “Correct” or “Match”. Though images which is classified into “Match” contain images of other categories, they can be recognized by conventional methods for a character image extracted. Therefore, the sum of “Correct” and “Match” shows the effectiveness of the proposed method.

Then, we consider the causes of “Miss”. 27% of Maru Gothic were classified into “Miss”. One of the reason is thresholds. When $T_2 = 0.75$ were employed instead of $T_2 = 0.85$, all categories were correctly extracted (i.e., 6 categories were “Correct” and the others were “Match”). Therefore, this problem can be solved by determining the proper thresholds for the images.

Also, “K” of Arial Black, “5”, “7”, “K” of Franklin were classified into “Miss”. An image of feature points extracted from the reference image of “K” is shown in Fig. 9. In the figure, too many feature points were extracted. This seems to be the main cause of the failure.

Font	Correct	Match	Miss
Arial Black	18 (29%)	43 (69%)	1 (2%)
Franklin	15 (24%)	44 (71%)	3 (5%)
Maru Gothic	22 (35%)	23 (37%)	17 (27%)

Table 1. The numbers of occurrences and their ratios for different fonts. The sum of ratios is not always 100% because of the ratios are rounded.

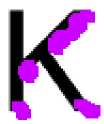


Figure 9. A cause of errors: too many feature points of the lower right corner were extracted.

4.2.2 Against isolated characters

To confirm the effectiveness of the proposed method against isolated characters, a photograph of a clock including numerals were used. Numerals of Arial were used as the reference images. $T_1 = 0.75$ and $T_2 = 0.89$ were used. Only “1” were “Match” and the others were “Correct”. Recognition result of “2” is shown in Fig. 10. In the picture, there are several “1”s and “2”s. However, they all were successfully recognized though they are a different font.

Segmentation of characters in the photo is very hard for conventional methods. This is a novelty of the proposed method.

4.2.3 Against touching characters

An image including touching characters [2] was used as the input image. “S” of Arial was used as the reference image. $T_1 = 0.75$ and $T_2 = 0.7$ were used. The size of the reference image was changed to fit larger “S” in the input image. The recognition result in Fig. 11 shows that “S” was correctly extracted (“Correct”). In addition, “e” was also extracted (“Match”).

In this experiment, since the size of the reference image was fit larger “S” in the input image, smaller “S”s were not extracted. Therefore, an invariant method for the size of the character image is required.

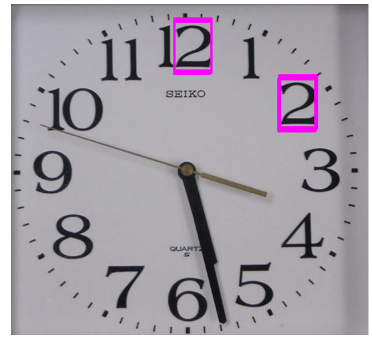


Figure 10. Recognition of isolated characters: the reference image is “2”.

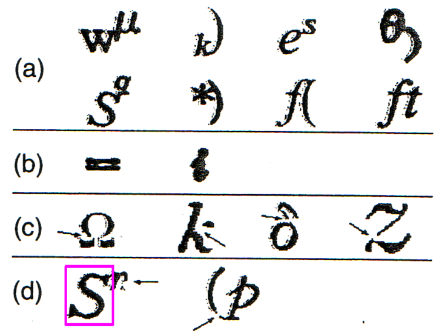


Figure 11. Recognition of touching characters: the reference image is “S”.

5. Conclusion

In this paper, we proposed a novel character recognition method which executes segmentation and recognition simultaneously. Experiments showed the effectiveness of the proposed method against (1)different fonts, (2)isolated characters, and (3)touching characters. Developing an automatic selection method of thresholds and an invariant method for the size of the input image are future works.

Acknowledgement This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (C), 16500096, and Young Scientists (B), 17700205, 2005.

References

[1] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.

[2] S. Uchida, A. Nomura, and M. Suzuki. Quantitative analysis of mathematical documents. *IEICE Technical Report*, PRMU2003-234, 2004. Written in Japanese.