An Extraction-based Stroke Matching Method for Handwritten Chinese Characters

[1*]*Sim Ao Yi* (沈鰲毅), [1]*Gan Chee Kim* (顏子鈞), *and* [1]*Jian-Jiun Ding* (丁建均)

[1] Department of Electrical Engineering,
National Taiwan University, Taipei, Taiwan,
[*]E-mail: b08901143@ntu.edu.tw

ABSTRACT

Stroke matching of handwritten Chinese characters plays a crucial role in a wide range of applications, including Optical Character Recognition (OCR) and cultural preservations. While having such importance, difficulties are often faced due to several factors: complex structures and combinations of stroke, individual writing style and ambiguity. With digital image processing algorithms advancing over time, the focus of this specific task has been shifted to gaining improvement in stability and robustness. Despite already having promising and advanced deep learning-based methods, many prefer a more traditional approach in the field of handwritten characters matching under consideration of various constraints and drawbacks. In this research work, we propose a stroke matching method featuring stroke extraction using contour information for handwritten Chinese characters. By having boundary points, corner points and segments information, cross sections in between opposite segments are established and stroke structures are validated. Next, the matching between input and standard characters is carried out using position, ratio, directional information and regional maps. With this proposed algorithm, we are able to quantify and measure the similarities of input handwritten Chinese characters to a set of ground truth characters in order to find corresponding stroke matches, even if the size and resolution of the input images may fluctuate.

Keywords: *contour detection, stroke extraction, stroke matching.*

1. INTRODUCTION

Chinese characters, like any other characters, are developed for the writing of its own language. Being in the form of logograms, they are considered as one of the oldest and most complex system of writing in the world. Due to their profound historic use and widespread current use throughout the Sinosphere, Chinese characters are also adopted in writing systems of different languages across multiple nations, including Japanese and Korean. In modern era, Chinese characters could be considered as a combination of a set of fundamental strokes, mostly composed of vertical or horizontal line segments. With different relative positions or orientation, these strokes form particular words. Moreover, differs from alphabetic writing systems, each logogram associates with an entire syllable and is semantic with few exceptions. Therefore, due to these distinctive properties, individual writing styles and habits often add to the complexities in handwritten Chinese character stroke detection and matching.

Stroke matching of handwritten Chinese characters indeed exerts a pivotal influence in OCR methods. Along with the global booming in deep learning researches and breakthroughs, a myriad of promising and assuring solutions for recognition and matching problems have been laid on the table. Since OCR is an important part of pattern recognition, deep learning contributes as it has good performance in pattern recognition field. For example, Chen, et al. proposed a convolutional neural network with adaptive local receptive field which resulted in significant improvement of performance in handwritten Chinese character recognition [2]. Yang, et al. proposed *DropSample*, a training method for deep convolutional neural networks (DCNNs) [3]. Zhong, et al. use the aid of directional feature maps and streamlined CNN to retain high accuracy [4].

In spite of having such powerful tools for performing pattern recognition tasks, the use of deep neural network in OCR often faces difficulties that restrain its abilities. Deep learning-based methods requires massive datasets which usually become one of the drawbacks in real-world situations, say in fields such as law, finance or criminal investigation. Besides, state-of-the-art CNNs appear to have deeper and larger networks, which incur huge computational cost and storage for more parameters. Though, many have been committed to resolve this specific issue [5, 6]. Alternative solutions to the problem are also proposed, for instance, sparse coding in compact modified quadratic discriminant function (MQDF) classifiers by Wei, et al [7].

In this paper, we propose a method in stroke matching of handwritten Chinese characters based on a rather traditional approach, featuring a stroke extraction algorithm using contour information. The algorithm integrated all levels of contour information, including boundaries, corners and segments. With all the features extracted, we can then establish cross sections in between tangentially parallel opposite segments to reconstruct character strokes. After gaining stroke information, our method then compare the strokes with a dataset of standard Chinese characters, in this case we used "楷體" (KaiTi). The comparisons are made using position, ratio and directional information, such that the strokes with maximum quantified similarities are considered as matching strokes.

2. PRIOR WORKS

Throughout the years, many algorithms were developed in order to surmount the difficulties posed in stroke extraction of handwritten Chinese characters. Most of these algorithms revolved around two main approaches: thinning or non-thinning based.

For thinning-based algorithms, the main constraint is the deformation and distortion in topology encountered by the end product after the skeletonization process. Researchers working on thinning-based method often tends to improvise in enhancing the topological robustness of the generated skeletons. For example, the maximum circle technique proposed by Liao and Huang [13], and implementation of it on top of a quadratic model for curve fitting employed by Wang et al [12].

On the other hand, there are researchers that worked around the constraints posed by skeletonization process in expectation of generate results with high accuracy and stability. Most of these works exploited a technique using the ambiguous junction between strokes, often referred as the cross-section-sequence graph (CSSG), including Huang et al.'s work [14].

In this work, we had surveyed carefully and dived deep into a lot of prior works in search of a practical and robust stroke extraction algorithm that is highly compatible with our stroke matching method. We came across an extraction method based on ambiguous zone detection proposed by Su et al. [10], also the run-length-coding-based approach by Fan and Wu [11]. In the end, we use an algorithm proposed by Lee and Wu [8], making use of accumulated information extracted from contour and corner detection.

3. STROKE EXTRACTION

Image contouring in image processing often refers to the extraction of the outline or borderline that is continuous. Contour is one of the most fundamental and essential information for shape analysis or object detection and recognition. By the same token, contour information plays an important role in our method for stroke extraction. We employ the algorithm proposed by Lee and Wu [8] which is exceptionally dependent on long contour segment that preserves stroke information instead of commonly used character skeletons. According to the assumptions in above method, technically different contour lines have a simple cycle form and could not exactly share the same pixel as part of their cycle.

Having the basic assumptions, one can proceed to perform preprocessing on the input handwritten Chinese character image. We make sure that the image is converted into binary image with character information conserved in regions with different values, i.e. foreground pixels having value one and the background regions having value zero, as shown in Figure 1 below.

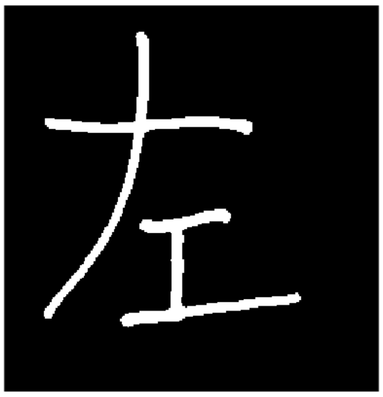


Fig. 1. An example of preprocessed binary image.

We first construct a 3×3 weighted kernel with specifically defined values such that for a targeted pixel, together with its eight-surrounding neighbors could form 256 combinations that are summed up to be a specific index value via element-wise multiplication with the weighted kernel, as illustrated in Figure 2b.

p_4	p_3	p_2
p_5	p_0	p_1
p_6	p_7	p_8

Fig. 2a. The targeted pixel and its 8-connected neighbors.

8	4	2
16	0	1
32	64	128

Fig. 2b. The 3×3 weighted kernel.

Since all 256 of the possible combinations are known beforehand, we define a lookup table that consist of instance indices from 0-255 representing the index value acquired through the dot product, with two main attributes: type and direction. The lookup table is as shown in Table 1. This lookup table is the preparation for contour tracing and the set of rules for assigning the attribute values are discussed.

For the type attribute, any pixel can be classified into three categories – interior, noise and boundary points with indices 1, 2 and 3 respectively. We define a set of neighbor index illustrated in Figure 3 and the definition of these categories is stated as below:

1. Interior pixel: for a pixel to have 8-neighbors of $(p_1 \cap p_3 \cap p_5 \cap p_7)$ & $(p_2 \cup p_4 \cup p_6 \cup p_8)$, it is considered as the interior content of a stroke which should be suppressed in boundary extraction.
2. Noise pixel: for a pixel to have 8-neighbors of $\left(p_i \cap p_{i+1} \cap \ldots p_{(i+n)\ mod\ 8}\right)$, where $0 \leq \mathrm{i} \leq 7$, $\mathrm{n} \geq 1$, it is considered as having a set of consecutive points. If the pixel satisfies one of the following two criteria, it is categorized as noise pixel:
   (a) The pixel does not have any consecutive points.
   (b) The pixel has exactly two sets of consecutive points but does not form a simple cycle as aforementioned.
3. Boundary pixel: for a pixel that is neither interior nor noise pixel, it is categorized as a boundary pixel.

For the direction attribute, we employ Freeman chain code to establish following direction for both the exterior and interior boundaries. For the exterior boundary, we abide by clockwise direction from the p_1 neighbor, vice versa for the interior boundary.

Table 1. A portion of the lookup table for contour tracing.

Index	Category	Direction
…	…	…
134	3	3
135	3	3
136	2	-
…	…	….

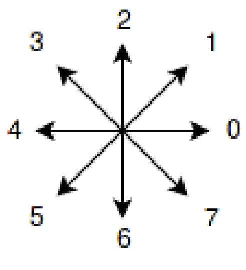


Fig. 3. Directions of the Freeman chain code.

3.1. Contour tracing

According to the rule set above, we can assort all the combinations into its own type and direction. For an arbitrary input image, we can use the lookup table to assign its type and extract the contour. However, throughout the process, whenever a noise pixel is encountered, this represents that the pixel is redundant and should be remove from the original image. After removing the noise pixel, its surrounding pixels must be redefined using the lookup table since and alteration has occurred. As the contour is extracted, we then obtain a 2D reference map where contour pixels have its index value while others remain null, for further tracing purposes.

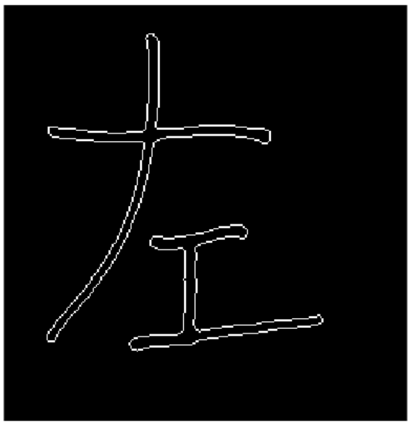


Fig. 4. An example of contour extraction result.

To trace the contour extracted in a specific direction, we first acknowledge that the strokes are often not connected and separated into different segments. By referring to the direction value using the lookup table, we group the contour pixels that form a simple cycle into segment and form a 2D directed map for the contour points.

3.2. Corner detection

Most of the time the intersection of strokes would likely has dramatic change in direction. Here we use a method exploiting the contour information extracted to detect these corner points. For each of the segments, we randomly choose a contour pixel as starting point and traverse throughout the segment according to the direction defined. Pixels in the front and back of the contour chain after a threshold value is obtained to calculate the included angle with current pixel as vertex, as shown in Figure 5. The threshold value here should be correlated with the image size and estimated stroke width, also optimized for arbitrary input image. The included angle is calculated using the arccosine function

$$cos^{-1}\frac{\overrightarrow{AB}\cdot\overrightarrow{CB}}{|\overrightarrow{AB}||\overrightarrow{CB}|}$$

With all the pixels have their included angle calculated, we then find the local minimum angle within the threshold as above to obtain the corner points. An example of the corner detection is shown in Figure 6.

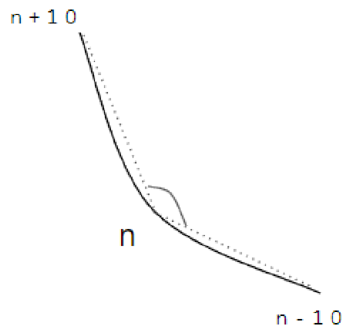


Fig. 5. Included angle form by front and end points of a contour chain.

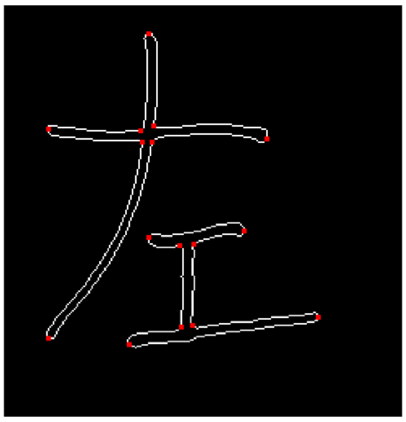


Fig. 6. An example of corners detection result.

3.3. Cross-section segments extraction

In this paper, we continue to follow the definition of segments in [8], where a segment is the fragment of the contour starting from a corner point and ends at another. The idea is to pair up different segments and form a valid stroke that is comparable and corresponds to the original character content. Using the directed map obtained in Section 3.1, we can form a set of segments with its respective length.

Since the character strokes have a certain stroke width and the edge or intersection region often consists of drastic changes in direction therefore resulting in corner points, it is reasonable that we assume that a pair of opposite contour segments preserves important information to recover a legitimate stroke.

To find the pairing segments, cross section is defined as a path between a pair of opposite segments, form by two points on the opposite segments which is normal in direction with almost parallel traversing direction. Here, we employ a sliding block method to calculate the tangential orientation of a target pixel by observing the direction of its surrounding pixels using the directed map. We then simply add 90 degrees to get its normal direction. A cross section sequence could be obtained for a segment using normal direction, and could have traverse across multiple different opposite segments. However, to save computational cost and memory, we only memoize the first cross section obtained for traversal in different opposite segments, i.e. if a segment has three different opposite segments, there would be three cross section recorded.

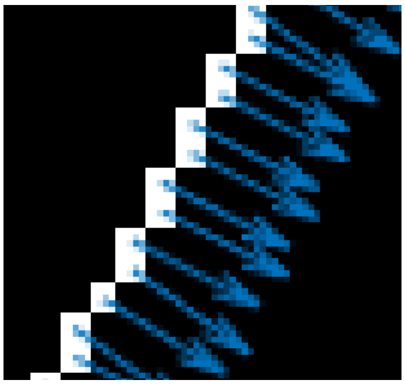


Fig. 7. Normal directions of different pixel along a given segment.

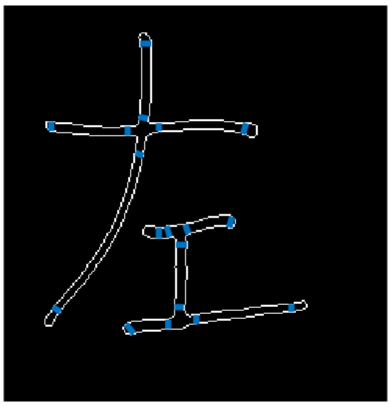


Fig. 8. The cross sections obtained from the pairing opposite segment sequence.

3.4. Segments pairing

We can pair the segments which cross sections have occurred, retrieve the meaningful segments and suppress the meaningless segments, as mentioned in [8]. For all paired segments, we take the longer stroke segment as the dominant segment to represent a particular stroke. If both segments in a pair has the same length, then arbitrarily choose one of the segments. The results of stroke extraction and pairing is shown in Figure 9.

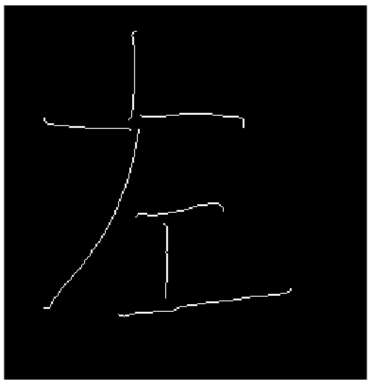


Fig. 9. The result of stroke extraction.

4. STROKE MATCHING

After the strokes have been extracted using method in Section 3, we then further use the gathered stroke information to acquire matching features between the input handwritten character and the standard character. This process is referred as stroke matching, as we apply several metrics to measure the similarities.

In order to quantize the similarities, we first need to extract the starting-, middle- and ending-points of the strokes as references. Here we strictly define these points of a particular stroke segment using a rule: that is the starting-point is the first point of the segment, while ending-point would be the last point of the segment and middle-point would be the point in between starting and ending-point located on that continuous stroke segment, such that the extraction of these points is always consistent.

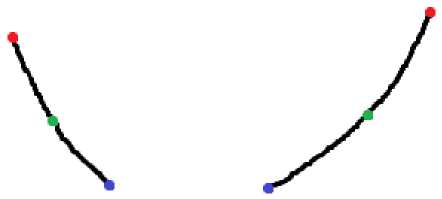


Fig. 10. Illustrations of starting, middle and ending-points for different strokes which are the red, green and blue points respectively.

4.1. Normalized locations of reference points

For the first metric, we simply compare the location of the reference points. This gives information on pixel coordinates of the strokes situated in both images, whereby the stroke segments in similar locations are more likely to be the corresponding strokes.

However, due to the fluctuation in sizes between the input image and the standard image, we consider the normalized location of the input Chinese characters according to the standard character size. Moreover, the discrepancy in individual writing styles or calligraphy together with orientation and placement of characters in the image often causes minor inaccuracies in representation of this metrics.

$$norm([row, col]) = ([row, col] ./ [H, W])$$

4.2. Proportion of strokes

Next up, we make use of the ratio of the pixel distributions of the reference segments to the total sum of distributed pixels in both characters. This gives details on the proportion of a specific stroke in regard to the whole character in the image. The information extracted further represents the structure, distribution of the character and of which is the dominant stroke.

$$ratio = \frac{total\ pixels\ of\ the\ stroke}{total\ sum\ of\ pixel}$$

Here, we consider using only extracted dominant stroke instead of the original images which contains the complete information of strokes. Thus, this property would then only reflect the length proportion of the characters. This metric is also highly dependent on the consistency between the handwritten styles and standard character and quite sensitive such that it is influenced by the alteration in direction of the strokes.

4.3. Direction of stroke structure

Third, we extract the directional information of the strokes in both characters. Through this information, we get how the strokes are behaving and their general orientation.

In a similar fashion to the tangent detection in Section 3.2, we use the tangent function to determine the tangent direction of the strokes. More specifically, we find directional deviation between the input and standard characters by simply take the difference in angle, the strokes with minimal difference have high possibilities to be the corresponding ones. Nonetheless, this metric is still constrained by shaky or messy, especially exaggerating tilted handwriting.

4.4. Information in-between strokes

As the metrics aforementioned in Section 4.1, 4.2 and 4.3 mostly consist of uncertainties caused by individual handwriting styles, the placement of character in the image, orientation or variating size of the characters. Therefore, taking the above metrics into consideration alone might not be the most robust system to measure the similarities between the input handwritten character and the standard characters. By the mean of this, we propose a method using contour lines around the targeted stroke to gain information in-between strokes such that the relative positions and relations of the strokes could be taken into account throughout the analysis process.

For a particular stroke in a character, we construct a set of two contour lines such that the three segmented regions have progressively decreasing weights $w = 1, 0.7, 0.3$ while drifting further from the targeted stroke, as illustrated in Fig. We set the shape of the contour line according to the contour of the targeted stroke segment and the contour interval is determine by $r = [\frac{H+W}{12}, \frac{H+W}{8}, \frac{H+W}{4}]$, where H and W are the height and weight of the image. We also determine that if the contour lines go out of bounds, the lines are instantly cut off. By summing up the weighted pixel values in each region, we could get the estimation of pixel distribution centering the targeted stroke.

By extracting this feature for all of the strokes in the character, we gain a relatively distinctive information on the character to such a degree aiding us differentiate and match the corresponding strokes.

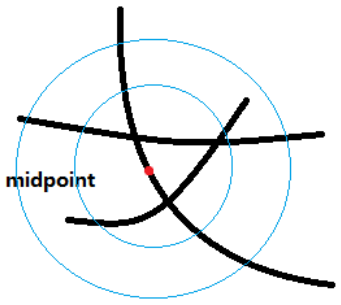


Fig. 11. An illustration of weighted contour map used in determining information in-between strokes.

4.5. Mean Squared Error evaluation

Having the features extracted in Section 4.1, 4.2 and 4.3, we use the mean squared error metrics to evaluate the quantified similarities within the strokes. For every stroke of the input handwritten character, we calculate its error by the difference in features from every other stroke of the standard character and find its mean. The minimum error obtained should be the corresponding stroke in terms of similarity. The MSE equation is as shown below:

$$\mathrm{MSE} = \frac{1}{\mathrm{N}} \sum_{i=1}^{N} (y\_input_i - y\_standard_i)^2$$

where N is the total number of features.

5. EXPERIMENTS AND RESULTS

To evaluate the effectiveness and robustness of our method, we run several tests on different Chinese character. We hereby demonstrate the course of matching the Chinese word ‘左’ with its ground truth counterpart in the font style named ‘標楷體’ (KaiTi). The train of thought in choosing Chinese characters as test case revolves around tackling the problems with stroke intersection, which is often encountered by the thinning-based method as deformation of skeletons are more likely to occur in junctions and ambiguous regions. Since the input handwritten character ‘左’ is already shown in above sections, here we only show the results of processing the standard character.

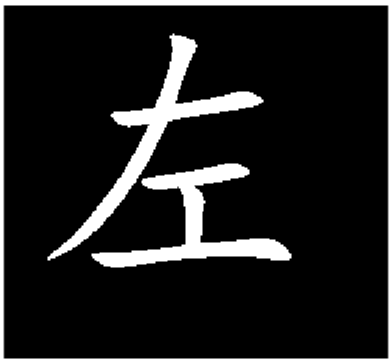


Fig. 12. Preprocessed binary image of standard ‘左’.

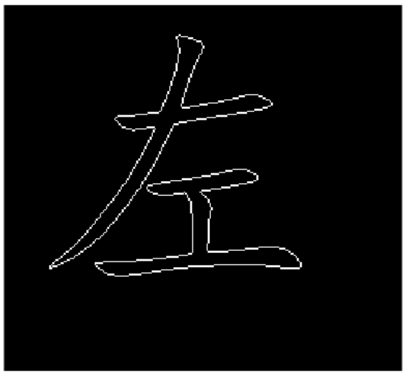


Fig. 13. Contour of the standard ‘左’.

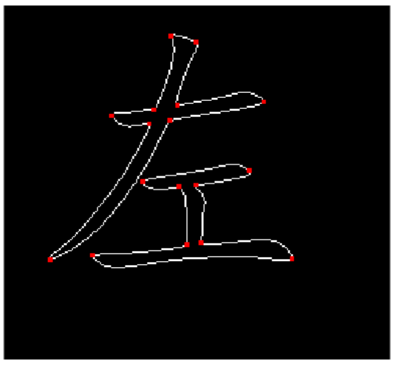


Fig. 14. Corner points of standard ‘左’.

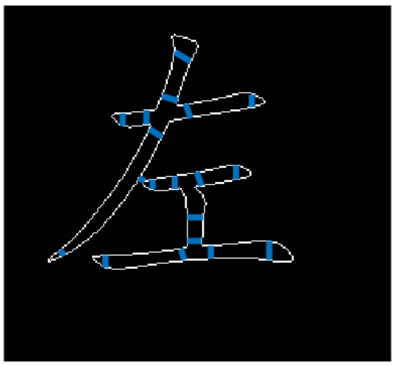


Fig. 15. Cross sections obtained in standard '左'.

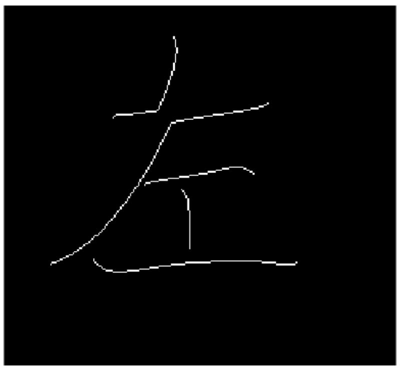


Fig. 16. Surviving dominant stroke of the standard '左'.

With the standard character dominant stroke extracted in Figure 16, we can then pass the strokes information of both standard and handwritten characters into the evaluation process. As a result, the strokes of '左' are matched quite accurately.

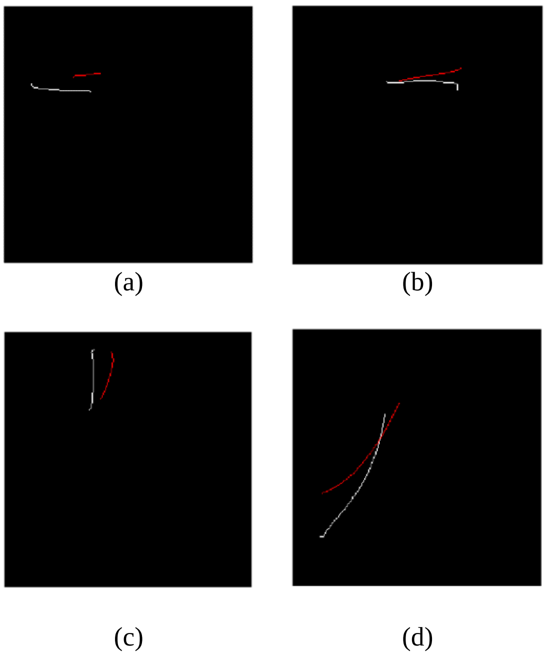
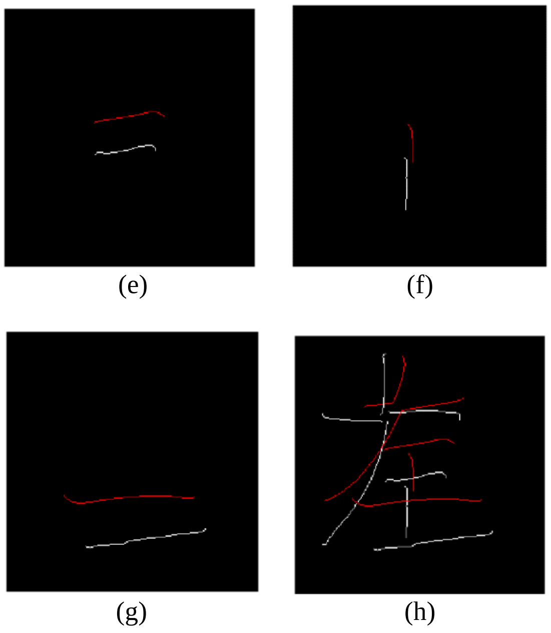


Fig. 16. (a)-(g) Corresponding matching strokes. (h) The entire matching characters.



6. DISCUSSIONS

In this research, we are aspired to find a robust solution for stroke extraction and matching based on preservation of contour information. However, there are some underlying factors and these worries could be the edge cases that cause decrease in the robustness of this method.

As we observed in the standard Chinese character written in KaiTi style, it has self-evidently larger stroke width and boxy silhouette, resulting in increase of meaningless segments and difficulties in pairing opposite segments. Furthermore, different styles of characters such as computer-generated funky style, typewriter and etc. have ambiguous and vaguely defined boundaries and corners, which could cause misjudge or misinterpretation. For an example, in Figure shows a error judgement of corner detection ultimately leads to continuity of two different strokes. This issue is also discussed in [8].

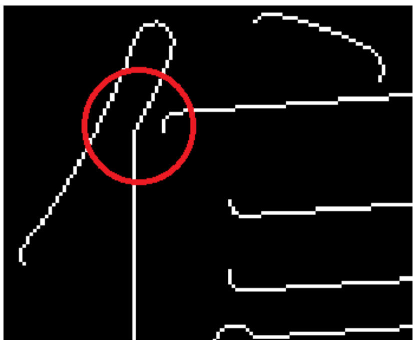


Fig. 16. Insufficient in detecting existing corner leads to wrong continuity.

6. CONCLUSION

We have developed a stroke matching method on top of the contour information-based stroke extraction method by Lee and Wu in [8]. By exploiting boundaries, corner points and cross sections, we are able to extract the strokes of an arbitrary input handwritten Chinese character. With the given information, we then proceed to perform feature matching on the strokes to find the corresponding legitimate strokes. In such way, we could work around the mainstream thinning-based algorithm which has disadvantage in preserving the topology of the skeletons, especially around junctions and ambiguous regions.

However, there are still underlying issues that are not yet solved and more deep diving are needed on this topic. Insufficiency in detecting smooth corners that are perceptible by human eyes is one of the main worries which may lead to incorrect continuity of strokes. In addition, individual writing styles or computer-generated fonts also caused deviation in the final result due to fluctuating stroke width and orientation.

7. REFERENCES

[1] A. Singh and A.S. Bist, "A wide scale survey on handwritten character recognition using machine learning." *Int J Comput Sci Eng* 7.6 (2019): 124-134.

[2] L. Chen, C.P. Wu, W. Fan, J. Sun and N. Satoshi, "Adaptive Local Receptive Field Convolutional Neural Networks for Handwritten Chinese Character Recognition." *Pattern Recognition: 6th Chinese Conference, CCPR 2014, Changsha, China, November 17-19, 2014. Proceedings, Part II 6.* Springer Berlin Heidelberg, 2014.

[3] W.X. Yang, L.W. Jin, D.C. Tao, Z.C. Xie and Z.Y. Feng, "*DropSample*: A new training method to enhance deep convolutional neural networks for large-scale unconstrained handwritten Chinese character recognition." *Pattern Recognition* 58 (2016): 190-203.

[4] W.X. Yang, L.W. Jin, D.C. Tao, Z.C. Xie and Z.Y. Feng, "*DropSample*: A new training method to enhance deep convolutional neural networks for large-scale unconstrained handwritten Chinese character recognition." *Pattern Recognition* 58 (2016): 190-203.

[5] X.F. Xiao, L.W. Jin, Y.F. Yang, W.X. Yang, J. Sun and T.H. Chang, "Building fast and compact convolutional neural networks for offline handwritten Chinese character recognition." *Pattern Recognition* 72 (2017): 72-81.

[6] Z.Y. Li, N.J. Teng, M. Jin and H.X. Lu, "Building efficient CNN architecture for offline handwritten Chinese character recognition." *International Journal on Document Analysis and Recognition (IJDAR)* 21 (2018): 233-240.

[7] X.H. Wei, S.J. Lu and Y. Lu, "Compact MQDF classifiers using sparse coding for handwritten Chinses character recognition." *Pattern Recognition* 76 (2018): 679-690.

[8] C.N. Lee and B.H. Wu, "A Chinese-character-stroke-extraction algorithm based on contour information." *Pattern Recognition* 31.6 (1998): 651-663.

[9] R. He and H. Yan, "Stroke extraction as pre-processing step to improve thinning results of Chinese characters." *Pattern Recognition Letters* 21.9 (2000): 817-825.

[10] Z.W. Su, Z.S. Cao and Y.Z. Wang, "Stroke extraction based on ambiguous zone detection: a preprocessing step to recover dynamic information from handwritten Chinese characters." *International Journal on Document Analysis and Recognition (IJDAR)* 12 (2009): 109-121.

[11] K.C. Fan and W.H. Wu, "A run-length-coding-based approach to stroke extraction of Chinese characters." *Pattern Recognition* 33.11 (2000): 1881-1895.

[12] A.B. Wang, K.C. Fan and J.S. Huang, "Recognition of handwritten Chinese characters by modified relaxation methods." *Image and Vision Computing* 12.8 (1994): 509-522.

[13] C.W. Liao, and J.S. Huang, "Stroke Segmentation by Bernstein-Bezier curve fitting." *Pattern Recognition* 23.5 (1990): 475-484.

[14] J.S. Huang, T.R. Jang and F. Chang, "Extracting strokes of a Chinese character based on cross-section-sequence graph." *J. Inform. Sci. Engng* 8 (1992): 512-524.