# A Kai Style Calligraphic Beautification Method for Handwriting Chinese Character[1]

Weiping Xia Lianwen Jin*
*School of Electronics and Information Engineering*
*South China University of Technology*
*xwpwlx@gmail.com* **lianwen.jin@gmail.com*

## Abstract

*In this paper, we propose a method that can beautify user's handwriting to the character of Kai style calligraphy. Structural and style features of Chinese character are summarized, and a model for representing character is proposed. Bezier curve with varying width is used for approximating user-input stroke segment. Corner shapes are modeled and utilized to beautify the corner points in the stroke. Rendering rules are designed based on the character features. Also, antialiasing technology is adopted to make the edge of the contour fine and smooth. Experimental results show the effectiveness of our system on beautifying the handwriting to Kai style calligraphy.*

## 1. Introduction

Calligraphy is an important part of Chinese traditional culture. Many font libraries with calligraphic style are widely used on prints and art designs. However, these fonts are of fixed shape that cannot accommodate to the user's input style. Digital ink technology [1] is used to beautify the user's input. However, the result brought by Digital Ink is somewhat monotonous and not similar enough to the real calligraphy.

A few studies have been taken out to beautify the user-input character or drawing. Our former work has implemented fusing the standard calligraphic style with the user's writing style by transfiguring the handwriting [2]. It can preserve the user's own individualities while applying the calligraphic style. However, the method needs several set of templates that store the contour and skeleton information of every character. Besides, character recognition is adopted, which makes it more complicated and less reliable. Songhua Xu et al proposed a system to generate new calligraphic style by blending existing calligraphic works [3]. But it cannot be applied to real-time user-interactive environment. A trajectory-based calligraphy simulation method is proposed in [4], which deforms user-input trajectories and results into Caoshu style calligraphy work. Yet the resulting style is somewhat monotonous and the overall architecture of a character is uncontrollable.

In this paper, we introduce a system that can beautify the handwriting to Kai style calligraphy. First, the structural and style feature of Chinese character are analyzed and a representing model of stroke is proposed. Then, the detail of beautifying curve segments and corners of the stroke is illustrated. After that, a set of rendering rules is designed and applied in constructing beautified characters. The experimental result shows the method we proposed is effective to beautify the user's handwriting to high quality Kai style calligraphy.

## 2. Features of Chinese characters

### 2.1. Structural feature of Chinese characters

On the basis of observation on large numbers of Chinese characters, we classify strokes into two categories according to their skeleton: (1) basic stroke, which contain only one segment that is overall flat; (2) compound stroke, which contain more than one segments that are joined at the corner position. Compound strokes can be formed by basic strokes. Strokes and their categories are shown in Fig.1.

### 2.2. Feature of Kai style calligraphy

The Kai font character is considered to be the most regular in all kinds of Chinese fonts. Its style is reflected by stroke segment and corner. The width of stroke is nearly constant during the writing process,

[1] This paper is partially sponsored by the following research projects: NSFC(no. U0735004, 60772116), GDNSF(no. 07118074)

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.59

with the middle of the stroke a little thinner. To the end of the stroke, the brush will be pulled up, and the stroke will become thinner gradually till a sharp end, that's called "Gou" in calligraphy terminology. When a corner is met, the brush will stay for a while and be given a little extra force so that a corner shape is formed. Some examples are given in the right column of Fig.1.

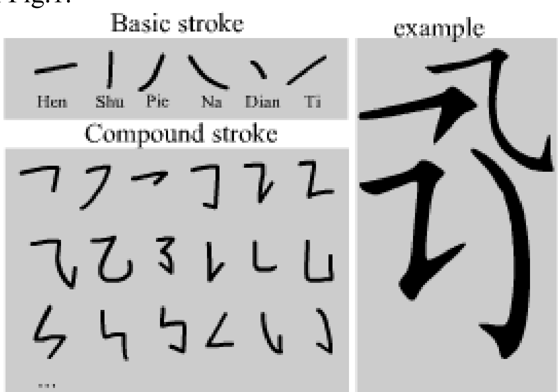


**Figure 1. Fundamental strokes**

It's obviously that the stroke is composed of several segments and corners. In general, the representing form of a stroke is given by Stroke=SC+S+{MC+S}+(EC). Where SC stands for the corner at the starting point of a stroke, MC the corner within the stroke, EC the corner at the ending point of the stroke and S the stroke segment. {x} means that x may not appear or repeat several times, while (x) means x appears zero or one time. Some examples are illustrated in Fig.2.

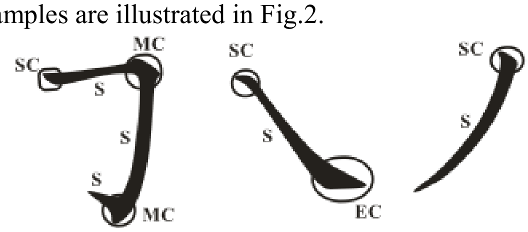


**Figure2. Stroke components**

We classify stroke segments into three categories by their width variation: (I) normal at the endpoints; while thinner in the middle; (II) normal at the starting point, and gradually thinner; (III) gradually become thick from zero width.

As illustrated in Fig.3, the type I segment is often part of 'Hen', 'Shu', and many compound strokes. Type II usually composes 'Pie', 'Ti', and end part of compound strokes, such as 'shu-gou', 'shu-wan-gou'. Type III is commonly contained in 'Na', 'Wo-gou', 'wan-gou'.

Nine patterns of corner are summarized in Table.1. Furthermore, they can be categorized into 3 categories: (1) point like corner; (2) single interface corner; (3) double-interface corner. Corner C1 is rendered as 'Dian', or point like 'Na' stroke. Single interface corners C2-C7 often appear at the starting or ending position of the stroke. Double-interface corners C8 and C9 are used as adapters for connecting one segment to another in different orientation, such as 'hen-zhe', 'shu-gou'.

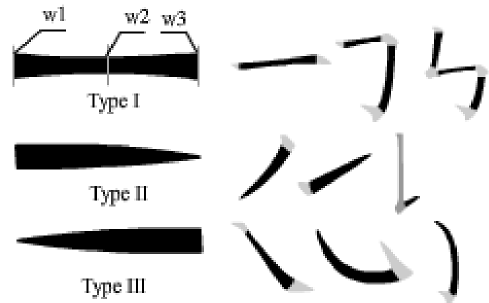


**Figure 3. Fundamental stroke segments**

**Table 1. Fundamental corners**

| | |
|---|---|
| Point | C1 |
| Single interface | C2 C3 C4 C5 C6 C7 |
| Double-interface | C8 C9 |

## 3. System architecture

The main flow chart of our proposed system is shown in Fig.4. The user's input trajectory, which is captured by mouse or touch screen, is first segmented into segments by corner points obtained in a corner detection procedure. Then, the segments are fitting into one type of the three segment types mentioned in section 2.2 according to rendering rules. Finally, the proper corner shape is generated at the corner position under the rendering rule and the beautified calligraphy is obtained.

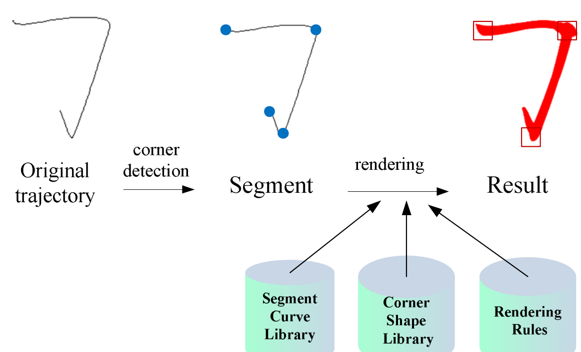


**Figure 4. System flow chart**

## 4. Stroke segment beautification

As shown in Fig.5, the stroke segment beautification procedure involves the following five steps.

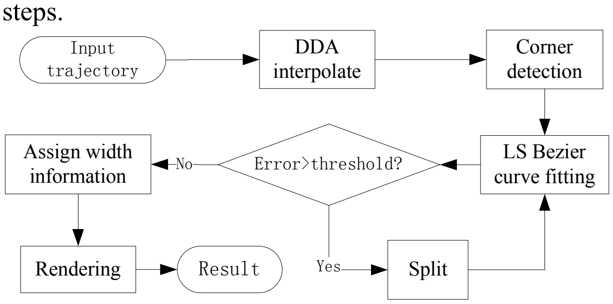


**Figure 5. Stroke beautification procedure**

Firstly, the input trajectory is interpolated using Digital Differential Analyzer method [5].

Secondly, corner detection is carried out on the interpolated curve. The Adaptive Bending Value method proposed by Wang [6] is adopted in our system. The curve will be divided into one or more segments by the corner points which include both endpoints.

Thirdly, each segment is fitted into Bezier curves using Least Square method [7]. The Bezier curves obtained is smoother, but may not so close to the original curve because of fitting error, as seen in Fig.6. The solution to this problem is to divide the curve into two parts at the maximum error point and fit each segment recursively until the error is small enough.

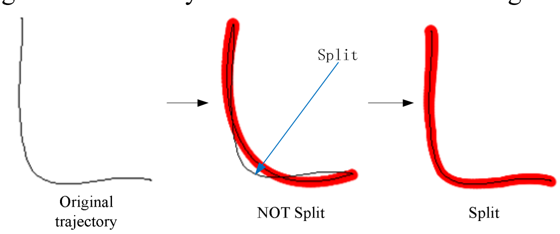


**Figure 6. Split and fit of curve**

Fourthly, assign the fitted curve with gradually-changing width shown in Fig.3. The values of the three parameters, *w1*, *w2* and *w3* are given in Table 2. Here we define *w* as the default width of the stroke, namely, the stroke width of the entire character. The width at the other points can be obtained using linear functions defined by *w1*, *w2* and *w3*.

**Table 2. Width for the stroke segment**

| Seg. type | w1 | w2 | w3 |
|---|---|---|---|
| I | w | 0.7w | w |
| II | w | 0.8w | 0 |
| III | 0 | 0.8w | w |

Finally, fill the stroke with circles whose diameter equals to the stroke width assigned in the fourth step. To avoid jagged appearance on the edge of the curve, the Wu method [8] is employed to draw antialised circle. The three types of rendering result are shown in Fig.7.

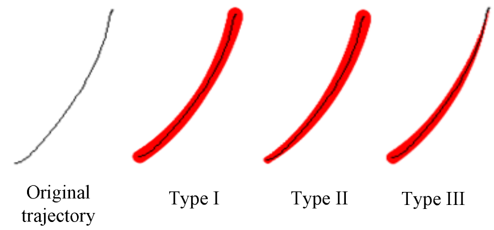


**Figure 7. Rendering result**

## 5. Corner beautification

The Bezier curve is also employed to model the contour of the corner. Let's take corner C4 given in Table.1 as an example to describe the general modeling principle. The contour can be modeled using 4 pieces of Bezier curve *AB*, *BC*, *CD* and *DA*, which are shown in Fig.8. The four control points of each curve are recorded so that to be used in the system. It's easy to scale, rotate and translate the model with Geometrical Transformation Technology [5].

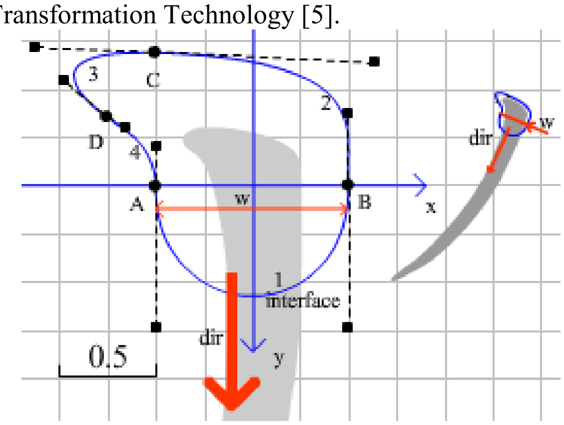


**Figure 8. Model of corner C4**

Since corner C4 is a single interface corner, two parameters are needed so that the corner can accommodate the stroke segment it connected: the size and orientation of the corner. We can enlarge the corner to size of *w*, which is the width of the stroke segment, and rotate it (*dir*-90) degrees, where *dir* is the orientation of the connected segment. Then, put the transformed model to the destination so that the corner shape is formed at the corner of the stroke.

To model corner C1 in Table.1, three parameters are to be introduced: the width along x and y axis respectively and its orientation. The rotating method is the same as the former one, while distinct scale factors for x and y axis should be applied as shown in Fig.9.

As for double-interface corner, which is shown in Fig.10, 4 parameters, the size and orientation of both interfaces are needed. The four points marked *A*, *B*, *C* and *D* are key to the model. Suppose the width of the segments with which the corner connected is *w1* and *w2*, and orientation is *dir1* and *dir2*. Here $\theta$ is the directional angle from vector 1 to 2. The sign of $\theta$ shows whether it's clockwise rotation from vector 1 to 2 or not. The coordinates of these points are given as follows:

$$\begin{cases} \alpha = \angle XOA = \angle XOB = \arctan(w/2/|OX|) \\ |OA| = |OC| = \sqrt{|OX|^2 + (w/2)^2} \end{cases},$$

$$\begin{cases} x_A = |OA| \cdot \cos(\theta - \alpha) \\ y_A = |OA| \cdot \sin(\theta - \alpha) \end{cases}, \quad \begin{cases} x_B = x_A + |AB| \cdot \cos(\theta + \pi) \\ y_B = y_A + |AB| \cdot \sin(\theta + \pi) \end{cases},$$

$$\begin{cases} x_C = |OC| \cdot \cos(\theta + \alpha) \\ y_C = |OC| \cdot \sin(\theta + \alpha) \end{cases}, \quad \begin{cases} x_D = x_C + |CD| \cdot \cos(\theta + \pi) \\ y_D = y_C + |CD| \cdot \sin(\theta + \pi) \end{cases}$$

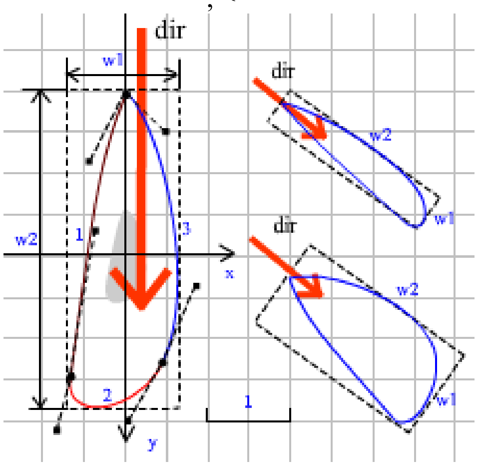


**Figure 9. Model of corner C1**

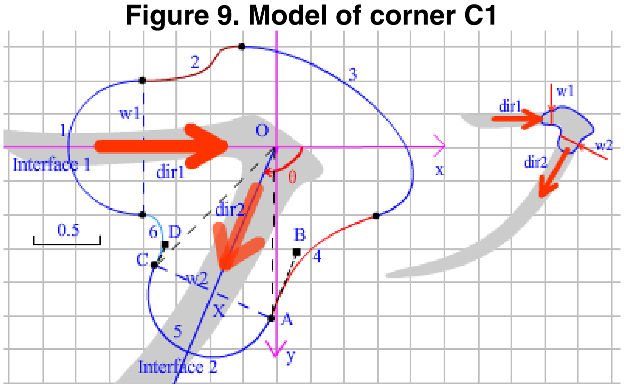


**Figure 10. Model of corner C8**

The length of ***AB*** and ***CD*** can be obtained from Fig.10, like 0.6 and 0.2 respectively. To accommodate the user's trajectory, model transformation is to be taken based on the parameters of interface 1. This procedure is the same as that of corner C4.

The flood-fill algorithm [5] can be adopted to fill the model so that the corner shape is formed.

## 6. Rendering rules

So far, we have presented stroke segment and corner modeling methods. The rule for determining which segment and/or corner shape to use is an important issue to address. We design a set of rules that can successfully achieve this goal.

### 6.1 Rule for basic strokes

We use the coordinate set as shown in Fig.8. Let's denote the orientation of the stroke as *dir*. The rules for selecting segment and corner type are given in Table3.

**Table 3. Rule for basic strokes**

| Stroke type | dir | Segments and corners to choose |
|---|---|---|
| Hen | [0,10]∪[350,360) | C2+I+C3 |
| Shu | [80,100] | Length>th1:C4 +I+C5<br>Length<th1: C4+II |
| Na | [10,80] | C6+III+C7 |
| Dian | Length < th2 | C1 |
| Others | N/A | C4+II |

The variable *th1* is relative to the stroke length that determines the ending point style of 'shu' stroke. It can be half the height of the character. *th2* is for determining whether the stroke is 'Dian' or not, and can be set to a small value such as 30. Some results that applied these rules are shown in Fig.11.

**Figure 11. Rendering result of basic strokes**

### 6.2 Rule for compound strokes

As for compound strokes, double-interface corners are used to join the two segments at corner position. The orientation parameter mentioned in Section 5 is determined by vectors formed by corner point and the point a short distance away, as shown in Fig.12.

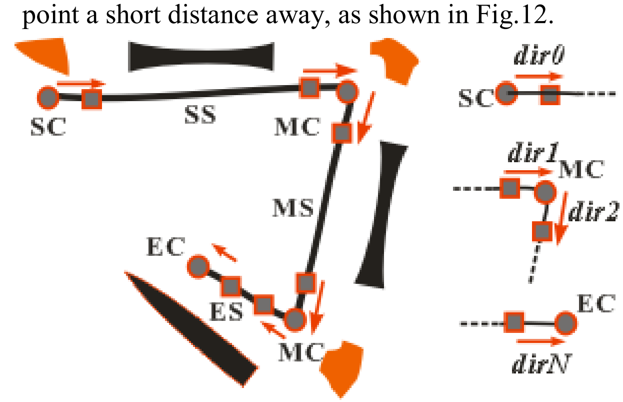


**Figure 12 Compound stroke rendering**

The rules for selecting segment type and corners for compound strokes are given in table 4 and table 5 respectively, where SS stands for segment at the start of the stroke, ES segment at the end of the stroke, and MS the other segments of the stroke.

**Table 4. Rule for choosing segment type**

| Segment type | Rule |
|---|---|
| I | SS: dir0∈[0, 10] ∪[350,360)<br>All MSs rendered as type I segment<br>ES: dirN∈[0,10]∪[350,360) |
| III | SS: dir0∈[30,70] |
| II | All the other situations |

**Table 5. Rule for choosing corner shape**

| Pattern | Rule |
|---|---|
| C2 | dir0∈[0,10]∪[350,360) |
| C4 | dir0∈[80,350) |
| C8 | 1. dir1∈[0,45] ∪[315,360) and dir2∈[80,180]<br>2. dir1∈[60,170] and dir2∈[0,20] ∪[290,360) |
| C9 | 1. dir1∈[45,145] and dir2∈[180,250]<br>2. dir1∈[0,60] ∪[240,360) and dir2∈[225,300] |

## 7. Experimental result

We applied our method on handwritings acquired both by writing on tablet PC (the first column in Fig.13) and the samples of online database for the "863 program of China" (the right three columns). The result shows our method can beautify the handwritings of various writing styles into Kai style characters.

Figure 13. Examples of beautification result

## 8. Conclusions

We proposed a method that can beautify user's handwriting to the Kai style calligraphy. A set of rules is designed based on the characteristic of Chinese Kai style calligraphy. Bezier curve is used for representing the stroke segment. 9 types of Corner shape are modeled in our system. Also, antialiasing technology is adopted to make the edge of the contour smooth and refined. Experimental results are given to illustrate the effectiveness of our method on beautifying the handwriting to Kai style calligraphy.

There are many potential applications for our system. First of all, it may serve as a tool for beautifying the handwriting in PC or tablet PC software. Our system can be considered as another version of Digital Ink Technology application, which can beautify and preserve user's handwriting. Second, the technology can be adopted in computer aided language learning system as a calligraphy practice function. With the help of our system, users can practice Kai style Calligraphy with a touch-panel or tablet PCs. Finally, the method can be used as preprocessing step of some character recognition [9]. The handwriting with Kai style may effectively improve the accuracy of the recognition.

## 9. References


[1] Heloise Hwawen Hse, A. Richard Newton, "Recognition and beautification of multi-stroke symbols in digital ink", Computers & Graphics, Vol 29, no. 4, Pages 533-546, 2005.

[2] Xinghua Zhu, Lianwen Jin, "Calligraphic Beautification of Handwritten Chinese Characters: A Patternized Approach to Handwriting Transfiguration", Proceedings of ICFHR2008, Aug. 2008, pp.135-140

[3] Songhua Xu, et al. "Automatic Generation of Artistic Chinese Calligraphy", IEEE Intelligent Systems, vol. 20(3): 32-39, May/June 2005.

[4] Mikihiro OHBA. Ougishi,, http://www.ne.jp/asahi/o/o/ougishi/

[5] Donald Hearn, M. Pauline Baker, "Computer Graphics, C version, 2nd Ed", Prentice Hall, 1997

[6] Wang M.J, Wu W.Y, Huang L.K, et al. Corner Detection Using Bending Value[J]. Patten Recognition Letters 199516(6):575-583

[7] Xiaohu Ma, Zhigeng Pan, "The Automatic Generation Algorithm of High Quality Outline Font Using Bezier Curve", Acta Automatica Sinica, Vol.20, No.1 Jan. 1994, pp.121-125

[8] Xiaolin Wu, Jon G. Rokne, "Double-step incremental generation of lines and circles", Computer Vision, Graphics, and Image Processing, pp. 331-344, Academic Press Professional, Inc. San Diego, CA, USA, 1987.

[9] Chenglin Liu, Ondrej Velek, Masaki Nakagawa, "Vector-to-Image Transformation of Character Patterns for On-line and Off-line Recognition", International Journal of Computer Processing of Oriental Languages, Vol.15, No.2 (2002) pp.187-209