Original Paper *Forma*, **16**, 357–366, 2001

Generation of Artistic Calligraphic Fonts Considering Character Structure

Koji YAMADA[1], Shuhei NISHIMURA[1], Tsuyoshi NAKAMURA[1], Lifeng HE[2] and Hidenori ITOH[1]

[1]*Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya 466-8555, Japan*
[2]*Aichi Prefectural University, Nagakute-cho, Aichi-gun, Aichi, 480-1198, Japan*

(Received June 15, 2001; Accepted January 30, 2002)

Keywords: Scratched Look, Blurred Look, Font Boldness Function, Character Structure, Ink Amount Function

Abstract. In this paper, we describe how to enhance the quality of brush-style-font expression, which are scratched look and blurred look, with regard to a form as a pattern of a Chinese character. The previous version of the system uses only a parameter similar to pen pressure, however actual calligraphic art requires more various factors in order to realize the expression. Our novel approach considers character structure, brush-drawing order and ink amount, and attempts to acquire effective brush-style expression.

1. Introduction

In these days, there have been several researches on computer-calligraphic art. For examples, (ZHANG *et al.*, 1984, 1985, 1986). And also, (YAMASAKI *et al.*, 1987) had studied and developed a CAI system for acquiring good writing skills in Japanese calligraphic art. Calligraphic fonts play an important role in some kind of softwares, such as a printing or supporting software for the creation of a greeting card or a New Year's card. Recently, these kinds of softwares employ a variety of calligraphic fonts and also use *Renmen-tai*, which is a flowing rounded style of writing with the letters joined together in order to generate artistic fonts that are much closer to actual calligraphic letters. In these trends, the processing of calligraphic fonts with scratched-look or blurred look expression is one of the advanced studies on font generation with the aim of artistic expressions. Only few studies have so far been made at generation of artistic Japanese calligraphic fonts with scratched look or blurred look. We have never seen the studies except (NAKAMURA *et al.*, 1993, 1994a, 1994b, 1995a, 1995b, 1996; MANO *et al.*, 1996), which dealt with only *Kaisho-tai* Chinese characters. Scratched look and blurred look mean "kasure" and "nijimi" in Japanese respectively. Scratched look appears on calligraphic characters when black ink sometimes does not reach paper from a writing brush: scratched look is a white region over strokes of a calligraphic character. Meanwhile, blurred look appears when black ink runs spread to paper from a writing brush: blurred look is blurred regions with ink.

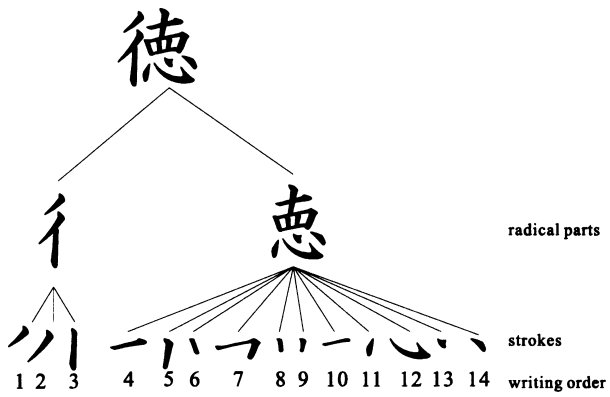


Fig. 1. Chinese character structure.

In actual calligraphy, we consider the following four factors which are going to decide the level of scratched look or blurred look: writing pressure, writing speed, writing technique and ink amount. Writing pressure affects calligraphic character's boldness. Even in progress of brush-writing a character, the pressure varies. The higher the pressure varies, the bolder the brush stroke becomes. Thus the writing pressure is considered to be related or similar to the boldness of an input font. The boldness was used as a parameter instead of the pressure (NAKAMURA *et al*., 1997).

Regarding writing speed and writing technique, it is difficult to estimate or derive them from a font shape or common knowledge. On the other hand, ink amount that is the amount of ink writing brush contains keeps on monotonic decreasing according as writing brush moves. From this feature, if either writing orders or its similar data is given, we will be able to assume it to be ink amount parameter.

Chinese characters have hierarchical structure. The 1st layer, which is a character itself, consists of radical parts which are called "bu-shu" in Japanese. The 2nd layer, which is a radical part, consists of strokes which are called "kaku" in Japanese (see Fig. 1). Each character has only one writing order restricted. Regarding the character "toku" illustrated in this figure, the number under the stroke indicates the writing order. A writing order is given to a sequence of strokes. Thus writing orders completely differ among characters. If we constructed the database regarding writing orders, the number of the data would be equal to the huge number of Chinese characters. In addition to that, it would be difficult to extract the exact strokes out of calligraphic fonts with aiming to give the information of writing orders to the font, and it might require a lot of processing time.

This paper considers a practical approach. We deal with a radical part as a unit for our processing. That is to say the database is constructed regarding the writing order of radical parts. The database we proposed and constructed in this study consists of only 10 basic pattern data. It is not huge compared with the total number of all Chinese characters. In addition to that, it is not so difficult to divide a calligraphic font into radical parts in comparison with stroke extraction, and it does not require a lot of processing time.

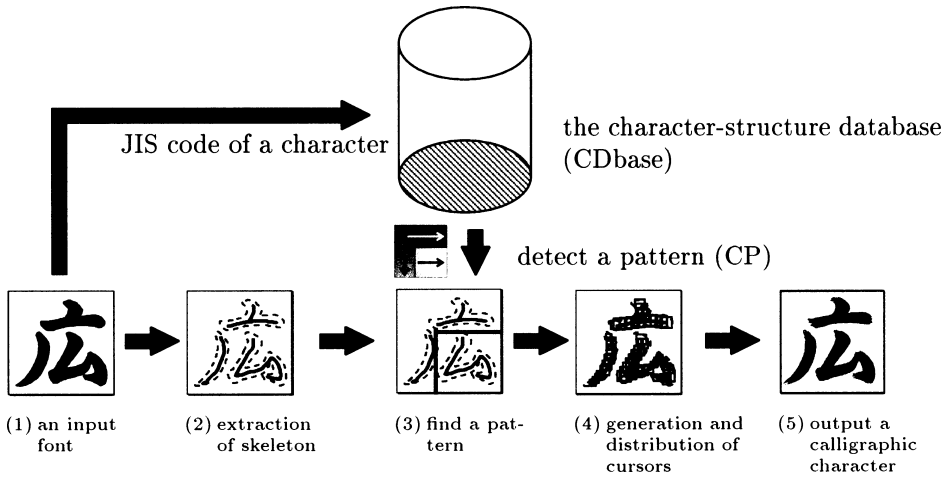


Fig. 2. Process overview.

In the previous system, boldness is only one parameter in order to decide the level of scratched-look or blurred-look expression. Thus for a certain font type, such as fonts with some constant boldness, the system does not work effectively. In our novel approach, we provide a database that has character-structure patterns and ink amount function. The information from the database, which indicates ink amount variation, enables to acquire more varieties of font appearance in comparison with the ones the previous system generates. In other words, the present system realizes more effective scratched-look and blurred-look expression with regard to a form as a pattern of a Chinese character by using both the boldness of an input font and the ink amount parameters.

2. System Overview

Figure 2 illustrates our process overview. First of all, in the stage (1), convert an input calligraphic font into a bit map font. Search the character-structure database (CDbase) and detect the character-structure pattern (CP) by using JIS code of the input font character as a key. In the stage (2), Hilditch's thinning algorithm (HILDITCH, 1969) extracts skeleton data of the font. In the next stage (3), fit CP in the skeleton data. This process gives the information of ink amount variation to the skeleton. In the stage (4), distribute and put brush cursors over the skeleton. In the last stage (5), output a font with scratched look and blurred look. In the stage (4), a cursor distributed on the skeleton consists of black or white dot (binary-data image), illustrated on the right in Fig. 3. This cursor is generated from a multi-value cursor that has values ranged from 0 to 255 using the following threshold function, $th(n, g_m(x, y))$ in Eq. (1).

$$th\big(n, g_m(x,y)\big) = th_o(n) \cdot g_m(x,y) + \alpha \tag{1}$$

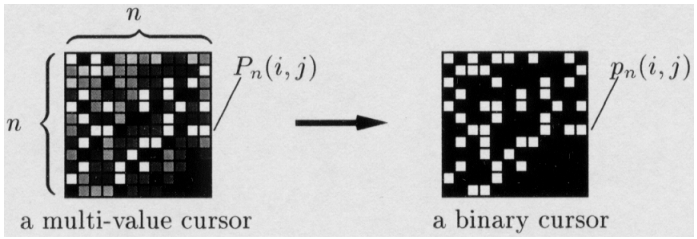


Fig. 3. Generation of binary cursors.

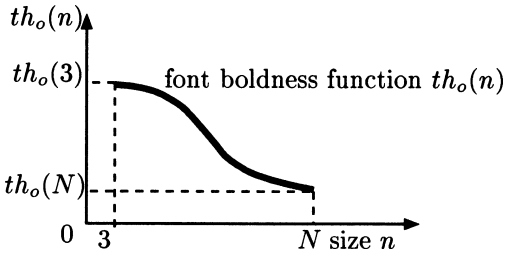


Fig. 4. A font boldness function $th_o(n)$.

$0 \le g_m(x, y) \le 1$
α is constant.

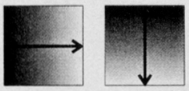
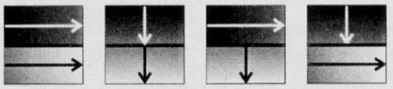
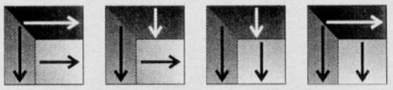
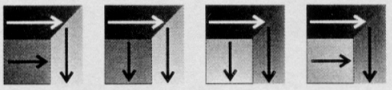
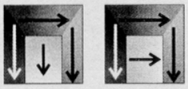
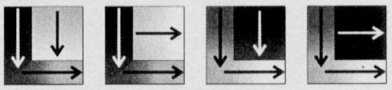
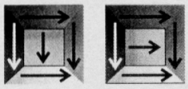
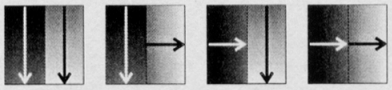
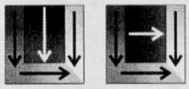
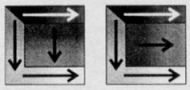
After the size of a multi-value cursor is appropriately scaled and adapted to boldness of the font at (x, y) coordinate of each skeleton pixel, a binary cursor that put at (x, y) coordinate is calculated by the following Eq. (2):

$$p_n(i,j) = \begin{cases} \textbf{if } P_n(i,j) > th\big(n, g_m(x,y)\big) & \textbf{then black} \\ \textbf{if } P_n(i,j) \le th\big(n, g_m(x,y)\big) & \textbf{then white} \end{cases} \tag{2}$$

$P_n(i, j)$: pixel value of a multi-value cursor
$p_n(i, j)$: pixel value of a binary cursor.

$P_n(i, j)$ indicates a pixel value of (i, j) coordinate on an $n \times n$ multi-value cursor, the left one in Fig. 3. $p_n(i, j)$ also indicates a pixel value of an $n \times n$ binary cursor, the right one in Fig. 3. If $P_n(i, j)$ is larger than $th(n, g_m(x, y))$, corresponding pixel value, $p_n(i, j)$, is given black, otherwise, white. It brings scratched-look or blurred-look expression to the font. Besides the white part area of scratched look tends to appear clearly as the number of white pixel is larger, which means $th(n, g_m(x, y))$ is larger. In contrast to that, as $th(n, g_m(x, y))$ is smaller, the part of scratched look does not tend to appear.

Table 1. Character-structure database (CDbase).

#	basic pattern	patterns of character structures (CP)	ex.
1			木 亜
2			恩 花
3			広 圧
4			式 気
5			開 風
6			通 延
7			国 囲
8			快 技
9			画 函
10			医 匡

$th_o(n)$ is a function illustrated in Fig. 4. x-axis indicates brush-cursor size n, and y-axis indicates $th_o(n)$ that is monotonic decreasing function. The cursor size corresponds to boldness of a font, so we call $th_o(n)$ font-boldness function. Thus $th_o(n)$ has a large value when n is small: the part of a font is thin. $g_m(x, y)$ is a function that indicates ink amount of the (x, y) coordinate, on CP m. This is called ink-amount function. The value of $g_m(x, y)$ is decided according to the writing order in CP m.

As mentioned above, $th(n, g_m(x, y))$ brings effective scratched-look expression: scratched look does not tend to appear so much when the part of a font is thick and the

writing order is closer to first one, whereas scratched look tends to clearly appear when the part of a font is thin and the writing order is closer to last one. If $g_m(x, y)$ =1:0 and $\alpha = 0$, it takes the same scratched-look and blurred-look processing as the previous system only depending on boldness of a font. In this paper, the description of blurred-look processing is omitted because it has no difference between the our present version and previous one of the system.

3. Character-Structure Database (CDbase)

3.1. Character-Structure Pattern (CP)

Chinese characters can be divided into some radical parts depending on their character features called "bu-shu". Some of characters have the same structures: they consist of the same radical parts or similar ones. In this paper, JIS code "kanji" character sets are classified to 10 basic patterns. In this case, a basic pattern consists of one or two radical parts. In addition to that, a basic pattern is re-classified to CPs that have a different ink amount function $g_m(x, y)$ according to the writing order and its direction. Each Chinese character necessarily corresponds to a CP.

Table 1 illustrates CDbase. The right column of Table 1 indicates character examples correspond to the each basic pattern. The gray scale level of a CP indicates values of $g_m(x, y)$. The darker gray scale level is closer to 0.0 and it is considered that it contains a lot of ink. Whereas, it is closer to 1.0 and it is considered that it contains small amount of ink. Arrows drawn on CPs denote that the ink amount decreases along the direction of arrows. In Table 1, arrows colored by either black or white exist, but it has no difference between them. The total sum of CPs are 202*.

In the case of a radical part, "Nyou", it corresponds basic pattern 6 in Table 1. The left two of CPs correspond "Ennyou" and the next two correspond "Shinnyou". Both "Ennyou" and "Shinnyou" take the same basic pattern, but CPs differ because they take different $g_m(x, y)$.

3.2. Adaptation to a variety of font types

We employed HG *Kaisho-tai* font type as sample font when constructing our CDbase. The size of radical parts slightly or terribly differs between HG *Kaisho-tai* font type and another even if both font types take the same character. We call "dividing line", the line to divide the character font into radical parts. The dividing line is shown in each basic pattern of Table 1. The location of dividing lines should be decided to adjust to a font type. We employ the method "Line Density Equalization" (YAMADA *et al.*, 1984) in order to solve this problem.

When an input font type is different from the sample font type HG *Kaisho-tai*, the line-density equalization normalizes an input font in terms of line density. Meanwhile sample

*In Table 1, some of CPs are omitted because the CDbase has total 202 CPs. It has no room to illustrate all of them in this paper. The omitted ones differ just regarding the area size of each individual radical part in a basic pattern. For example, the upper radical part of "花" is small area and the lower is larger than the upper. Regarding "恩" it is opposite.

the previous system　　the present system 1　　the present system 2

Fig. 5. Output examples.

font is also normalized in the same way as the input font. It has nothing to say that both of them deal with the same character for normalization. f_I and f_S are normalizing functions for an input font and sample font respectively. d_I and d_S indicate positions of dividing lines for an input font and sample font respectively. The purpose here is to detect d_I. In the normalization processing, the system handles images after thinning processing of sample and input font characters.

After the normalization processing, both fonts form similar shape. This property lets us guess $f_I(d_I)$ to be similar to $f_S(d_S)$. Of course $f_S(d_S)$ is known at present. Besides there is the inverse transformation for f_I. Thus d_I is described as Eq. (3). $g_m(x, y)$ for sample font is also transformed using Eq. (3), and $g_m(x, y)$ for an input font is acquired.

$$d_I = f_I^{-1}\left(f_S(d_S)\right). \tag{3}$$

4. Output Examples

Figure 5 illustrates output examples. From the left, examples of the previous system, examples 1 and 2 of the present system. The center examples of the system are acquired as a result of scratched-look processing only. The right examples are acquired as a result of both scratched-look and blurred-look processing. The output examples of this system well represent ink decreasing according to the writing order. In this respect, it seems that the system works effectively.

5. Conclusions

Table 2 illustrates the number of characters with wrong dividing lines for each basic pattern. We checked all the Chinese character of the first level character set of JIS code (2,965 characters). The row of "the number of JIS characters" indicates the number of character included in each basic pattern. The following two rows, *Gyousho-tai* and *Hagoromo-tai*, indicate the number of Chinese characters with wrong dividing lines*. The ratio of the characters with wrong dividing lines is less than 4% in the first level character set of JIS code. The line density equalization almost works well except some cases.

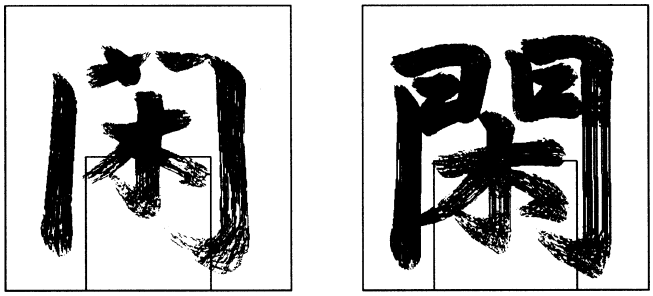
From this table, 6th pattern of *Gyousho-tai* from the left of the table indicates large number of failures. This is because *Gyousho-tai* is a script style, the deformation of "mongamae", which is a kind of the radicals, is too excess. Output examples of this case are illustrated in Fig. 6.

The miscalculation of line density equalization causes wrong dividing lines. The largest number of failures is the pattern divided vertically, the second basic pattern from the right in Table 2. In this case, it is only possible in this system to divide a character straightly only. Thus some character cannot be divided into two areas completely. As a result, unnatural scratched-look expression tends to appeared.

In this paper, patterns of character structures are clustered by rough writing orders that are given in one area unit. The total number of CPs in the database is 202, this figure is quite less than 6,535, which is the total number of Chinese characters of JIS character set. Besides in regard to calculation of dividing lines, it is relatively easy processing and it does not require so much calculation time because the number of divided areas is 2 areas maximum.

We targeted enhanced scratched-look or blurred-look expression using font boldness function and ink amount function. Figure 5 told that examples of this system have more variety to scratched-look and blurred-look expression than the case of the previous system. Thus our proposed method seems to be one of effective method, we think. But regarding to some font types that have completely different features from the sample font, such as

*The criterion of wrong dividing lines is that the another area of the radical parts includes wrong pixels more than 10% of the total skeleton pixels.



Gyousho-tai *Hagoromo-tai*

Fig. 6. Failure cases.

Table 2. The number of calligraphic characters with a wrong dividing line.

basic pattern										
the number of JIS characters	535	8	501	116	9	27	86	12	1667	4
Gyousho-tai				4		6	11		43	
Hagoromo-tai						2	9		97	

Sousho-tai, this system will be difficult to handle them. We have to discuss and solve this problem.

REFERENCES

HILDITCH, C. (1969) Linear skeleton from square cupboards, *Machine Intelligence*, **6**, 403–420.

MANO, J., NAKAMURA, T., SEKI, H. and ITOH, H. (1996) A scratched-look expression of calligraphy characters using renormalization group, in *International Fuzzy System and Intelligent Control Conference*, pp. 93–102.

NAKAMURA, T., ITOH, H., SEKI, H. and LAW, T. (1993) A writing system for brush characters using neural recognition and fuzzy interpretation, in *International Joint Conference on Neural Networks*, pp. 2901–2904.

NAKAMURA, T., KURODA, T., ITOH, H. and SEKI, H. (1994a) A calligraphy system based on analyzing user writing speed, in *The 3rd International Conference on Fuzzy Logic, Neural Nets and Soft Computing*, pp. 189–190.

NAKAMURA, T., KURODA, T., ITOH, H. and SEKI, H. (1994b) Fuzzy-based writing system for acquiring good writing skill of brush characters based on the analysis of writing speed, in *The 3rd Pacific Rim International Conference on Artificial Intelligence*, Vol. 2, pp. 822–827.

NAKAMURA, T., KURODA, T., ITOH, H. and SEKI, H. (1995a) A calligraphy system based on fuzzy-evaluation of the writing speed, *Journal of Japan Society for Fuzzy Theory and Systems*, **7**, No. 2, 371–379.

NAKAMURA, T., NOZAKI, K., SEKI, H. and ITOH, H. (1995b) A fuzzy-based calligraphy system using fractals, in *The 3rd European Congress on Intelligent Techniques and Soft Computing*, pp. 1440–1444.

NAKAMURA, T., MATSUSHITA, M., SEKI, H. and ITOH, H. (1996) A scratch-expression of calligraphy characters using fractals, *Journal of Japan Society for Fuzzy Theory and Systems*, **8**, No. 3, 558–566.

NAKAMURA, T., MANO, J., SEKI, H. and ITOH, H. (1997) A system for generating various calligraphy characters

with scratched look or blurred look, *The Transaction of Information Processing Society of Japan*, **38**, No. 5, 1008–1015.

YAMADA, H., SAITO, T. and YAMAMOTO, K. (1984) Line density equalization—a nonlinear normalization for correlation method—, *The Transaction of the Institute of Electronics, Information and Communication engineers*, **J67-D**, No. 11, 1379–1383.

YAMASAKI, T., YAMAMOTO, M. and INOKUCHI, S. (1987) Cai system for acquiring good writing skills based on the analysis of pen speed, *Trans. IEICE Japan*, **J70-D**, No. 11, 2071–2076.

ZHANG, X., SANADA, H. and TEZUKA, Y. (1984) Forming square-styled brush written Chinese characters with a computer, *Trans. IEICE Japan*, **J67-D**, No. 5, 599– 606.

ZHANG, X., SANADA, H. and TEZUKA, Y. (1985) Forming rei-styled Chinese characters with the method of hierarchical synthesis by analysis, *Trans. IEICE Japan*, **J68-D**, No. 8, 1489–1496.

ZHANG, X., SANADA, H. and TEZUKA, Y. (1986) A proposal of brush-touch pattern suitable for generating various styles of characters with a computer, *Trans. IEICE Japan*, **J69-D**, No. 6, 885–892.