International Journal of Computer Processing of Oriental Languages, Vol. 15, No. 2 (2002) 187–209
© Chinese Language Computer Society & World Scientific Publishing Company

Vector-to-Image Transformation of Character Patterns for On-line and Off-line Recognition

ONDŘEJ VELEK* AND MASAKI NAKAGAWA
Tokyo University of Agriculture & Technology,
2-24-16 Naka-cho, Koganei-shi, Tokyo 184-8588, Japan
**velek@hands.ei.tuat.ac.jp*

CHENG-LIN LIU
Central Research Laboratory, Hitachi Ltd.,
1-280 Higashi-koigakubo, Kokubunji-shi, Tokyo 185-8601, Japan

This paper proposes a method to generate realistic character images from on-line patterns. From the pen trajectory of an on-line pattern, the proposed method generates images of various stroke shapes using four painting modes: constant line mode, proportional mode and two calligraphic modes. Particularly, the calligraphic modes combine the pen trajectory with real stroke-shape images so that the generated images resemble the characters produced with brush pen. In the calligraphic painting mode based on primitive stroke identification (PSI), the strokes of on-line patterns are classified into different classes and each class of strokes is painted with the corresponding stroke shape template; while in the calligraphic painting mode based on stroke component classification (SCC), each stroke is decomposed into ending, bending and connecting parts, and each part is painted with a stroke shape template. Decomposing strokes into parts is helpful to deal with connected strokes in cursive writing. Our method of image transformation serves two purposes: supplying image samples for off-line recognition, and application of off-line recognition methods to on-line recognition. The experimental results show that our methods are well suited for both purposes. We show that calligraphic painting mode is appropriate for off-line recognition while constant line mode and proportional mode are appropriate for on-line recognition.

Keywords: Handwritten Character Recognition; Pattern Transformation; Calligraphic Painting Mode; Primitive Stroke Identification; Stroke Component Classification.

1. Introduction

A large database of samples is essential for designing high performance classifiers for character recognition. Collecting large numbers of character images from documents or directly from writers is very expensive and time-consuming, especially when the number of character categories is large, such as for Chinese or Japanese character recognition [1,2]. A way to circumvent this problem is to generate samples artificially by generative models [3-6] or modifying existing images [7,8]. In generative models, each character is represented in a relational structure of strokes or sub-characters and the structure is subject to deformations under certain constraints to generate pseudo-character images. The other possibility is to generate new images by transforming or deforming existing real character images.

In this paper, we utilize on-line databases collected primarily for on-line character recognizer design and benchmarking [9-12] in order to generate artificial character images. The motivation of generating character images from on-line patterns originates from two applications: integrating off-line recognition methods into on-line character recognition [14], and providing image samples for classifier design of off-line recognition. In the latter case, the generated images should simulate the actual situation of writing with various pen instruments. The proposed method in this paper aims to generate realistic character images from on-line patterns for off-line character recognition.

The proposed method can generate numerous images with various stroke shapes from a single on-line pattern using four painting modes: constant line mode, proportional mode, and two calligraphic modes. In calligraphic modes, we use the calligraphic knowledge of Chinese character writing to generate character images that look as if they were produced with brush pen. The "brush-style" samples not only enrich the variability of image samples, but also particularly benefit the postal address recognition of characters written with brush pens, which are popular in Japanese greeting cards.

From on-line patterns, we can also deform the trajectories to generate distorted samples and combine it with our approach to multiply a number of generated patterns.

In this paper we focus on generating character images from on-line patterns; i.e., vector-to-image pattern transformation. This is typically accomplished by drawing the trajectory vectors either with a constant line width or a line width proportional to the writing speed. We call these two painting modes constant line mode and proportional mode, respectively. We present two additional calligraphic painting modes that combine the pen trajectory with stroke shape templates to generate more realistic character images. The stroke shape templates are extracted from real stroke images scanned from patterns written with a brush. By using different stroke shapes written by different writing tools, we can generate many realistic off-line images from the same on-line pattern. This approach is especially useful for generating characters with complex stroke shapes, such as the characters produced with a brush pen.

To test the effects of our generated images on character classification, we made several experiments for on-line and off-line recognition. For off-line recognition, we use the generated images as training samples of a character classifier and test the classification performance on generated images and real off-line samples. For on-line recognition, we transform the on-line patterns into off-line images and use off-line methods for recognition. We will show that the images generated by calligraphic painting modes are more effective for off-line recognition, while the constant line mode and proportional mode are appropriate for on-line recognition.

The rest of this paper is organized as follows: Section 2 introduces the on-line databases that we used, Section 3 describes the painting modes for generating off-line images from on-line patterns, Section 4 presents the experimental results, and Section 5 provides the concluding remarks.

2. On-Line Databases

In recent years, two large databases of on-line characters have been collected in Nakagawa Lab of the Tokyo University of Agriculture & Technology [10,11,12], which are now publicly available to the on-line character recognition community. The two databases are called “Kuchibue” and “Nakayosi” respectively. It is noteworthy that there is no intersection between the writers of these two databases. In total, there are over 3 million patterns in both databases written by 120 writers (Kuchibue) and 163 writers (Nakayosi), respectively. Using the generation method proposed in this paper, a huge number of character images can be generated in various qualities for off-line recognition. The databases were collected under different conditions with 50 × 50 point sampling boxes. From these patterns, captured with relatively low-resolution, we generated bitmaps in high quality. The generated resolution was 96 × 96 points for the patterns used in recognition experiments and 500 × 500 points for the figures shown in this article.

Table 1. Statistics of Kanji character databases.

Database	Format	#Categories	#Writers	#Pattern		
				Per category	Per writer	Total
Kuchibue	vector	3356	120	120−47760	11962	1435440
Nakayosi	vector	4438	163	163−45314	10403	1695689
Etl9B	bitmap	3036	4000	200	N/A	607200
JEITA-HP	bitmap	3214	580	580−1160	3306	1917480

Both these on-line databases have the advantage that they account better for the variability in practice because the characters were written fluently in sentences without any style restrictions and their character shapes are more natural compared to most off-line databases, such as ETL9B. The statistics of the databases are given in Table 1 where they are also compared with other off-line databases, which will be described later in an experimental Section 4.1.

3. Generating Off-Line Images

3.1. Constant line and proportional painting modes

The character images of constant line mode are obtained simply by connecting the sampled points of a pen trajectory with lines of uniform width. These images can be directly used for off-line recognition. Also, multiple images can be generated easily from a single on-line pattern by using a different stroke width for each image. In practice, however, the stroke width of handwritten characters is rarely uniform. It varies depending on the writing instrument and the writing speed.

The proportional painting mode generates character images by assuming that the stroke width depends on the writing speed, which is either directly available in on-line patterns or can be recovered from the distances between two consecutive points. In proportional mode, the stroke shape can be controlled by two parameters, namely, the minimum width and the range of width. Figure 1 shows some examples of generated images, where the left four columns are painted in constant line mode and the right two columns are in proportional mode.

3.2. Calligraphic painting mode based on primitive stroke identification (PSI)

The aim of calligraphic painting mode is to generate character images that look as if they were written with actual pens, especially with brush pens. We achieve this goal by combining the pen trajectories of on-line patterns with the stroke shape patterns stored in a library. We compile our stroke shape pattern library by scanning the stroke images produced with actual brush pen and extracting the trajectory independent shape information. By combining the position and orientation of on-line trajectories with the shape information of the stroke-shape-pattern library, the generated images well resemble the characters written by actual pens. The stroke shape pattern library has several stroke types. The particular stroke type needed for drawing is determined by a stroke type

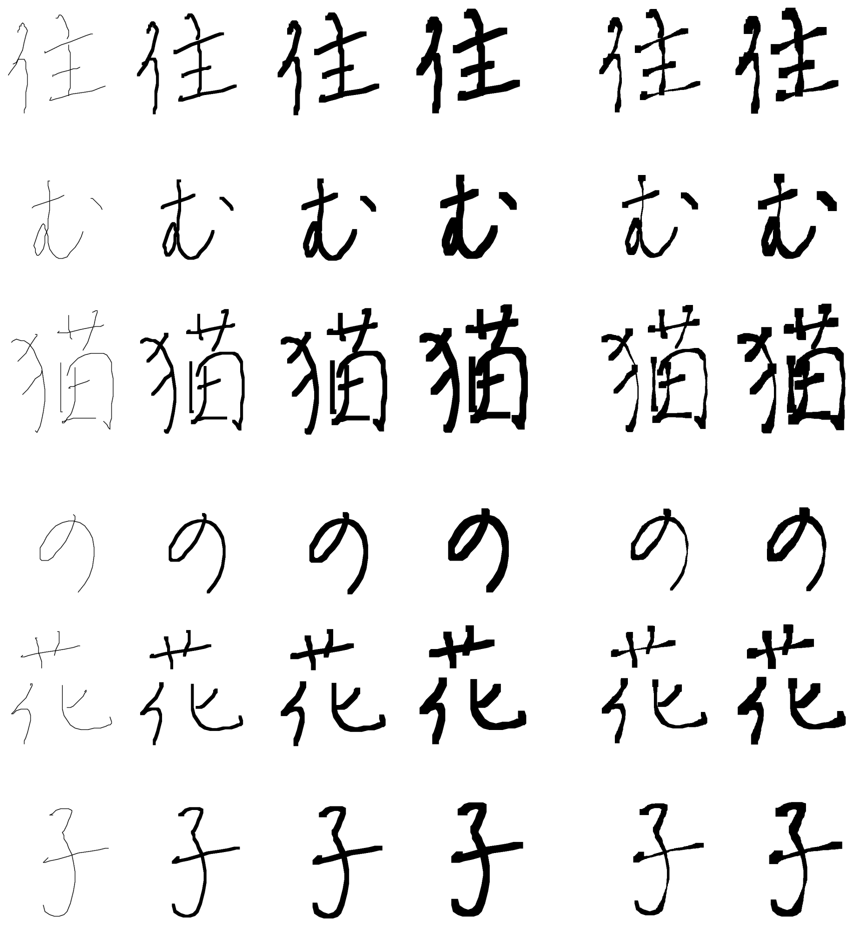
Figure 1. Examples of constant line and proportional painting modes.

recognition procedure on the on-line trajectory. The on-line strokes whose types are reliably recognized are combined with the corresponding stroke shape patterns, whereas those that are not recognized are combined with a default stroke shape pattern generated by a proportional painting mode or a pre-specified stroke shape pattern from the library.

3.2.1. Building the stroke shape pattern library

To build the stroke shape pattern library, we first scan real stroke images produced by human writers with a brush pen, as shown in Figure 2. The normalization process then extracts the stroke shapes for our library and discards any other information about the writing trajectory.

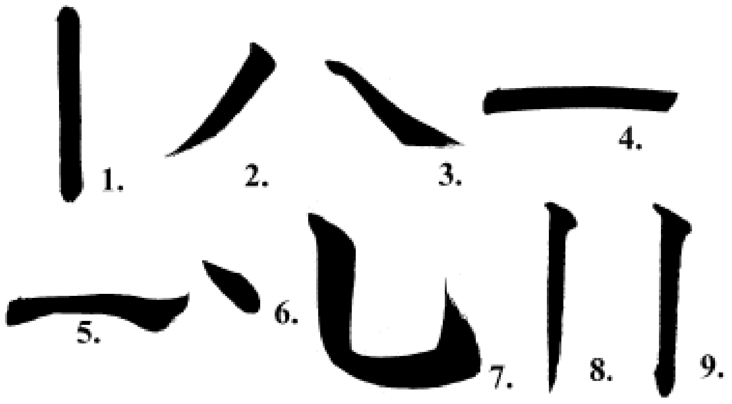

Figure 2. Stroke images produced with brush pen.

As shown in Figure 3-1, a trajectory is manually drawn into the scanned stroke image. Then this writing trajectory (Figure 3-2) is smoothed. For each point of this trajectory we construct the normal vector and compute the local stroke width, which is illustrated in Figure 4. The reason that we smooth the trajectory is that without smoothing the extracted vectors will be very unstable, as shown in Figure 3-3.

While Figure 3-3 was generated directly without smoothing, the more realistic Figure 3-4 was derived from the smoothed trajectory. In the final step, the outline is also smoothed, as shown in Figure 3-5.

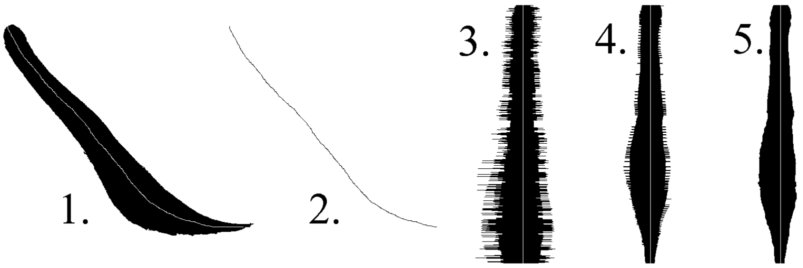

Figure 3. Normalized stroke patterns.

The normal vectors are classified into left vectors and right vectors, to record the width on both sides of the stroke. The library stroke shapes corresponding to the strokes in Figure 2 are shown in Figure 5. Our stroke shape library does not contain the full, normalized stroke bitmaps, but only left vectors and right vectors as shown on the left-hand side of Figure 4.

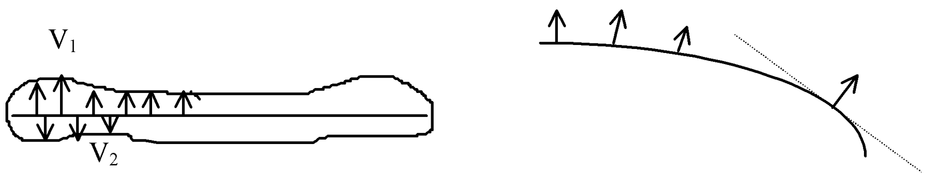

Figure 4. Painting process in calligraphic mode.

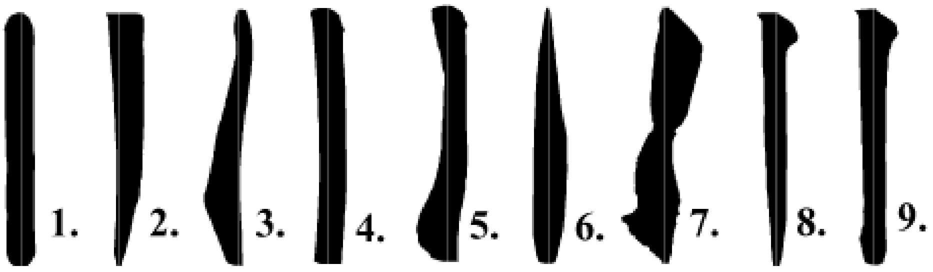

Figure 5. Stroke shape patterns from Figure 2 after normalization.

3.2.2. Drawing strokes

We transform on-line strokes into off-line images by combining their shape-independent trajectories with the trajectory-independent stroke shape information. In Figure 4, the left graph shows the normal vectors of a stroke pattern, and the right graph shows the trajectory of an on-line stroke. We draw a left vector and a right vector for each point of the trajectory. The heads of the vectors define the contour of the stroke. A post-processing step fills this contour black and yields the final shape of the stroke, as shown in Figure 6.

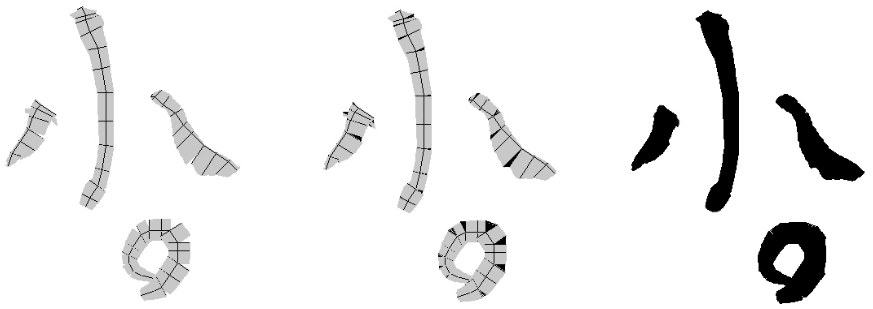

Figure 6. Examples painted in calligraphic painting mode.

In calligraphic mode, we control the stroke width and proportionality to generate various character images. Some calligraphic images of variable stroke width and proportionality are shown in Figure 7.

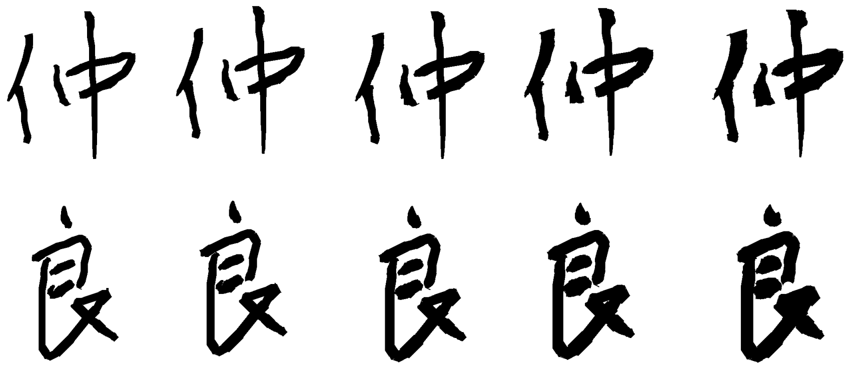

Figure 7. Calligraphic images with variable stroke width.

3.2.3. Classifying stroke types

To assign a proper stroke shape pattern from the library to each on-line stroke is not trivial. We have used a simple, but effective strategy to accomplish this task. We utilize information about the trajectory's convexity, concavity, and orientation to detect stroke types with high confidence. The recognized stroke types are combined with the corresponding stroke shape patterns, while the strokes of uncertain type are combined with a default shape pattern.

We classify on-line strokes into short strokes, line strokes, two-line strokes, and multi-line strokes. The short strokes are easily detected by checking their length. We set the threshold for the length of short strokes to 15 points given a character size of 100 × 100 points. A line stroke is basically a straight ink trace and is defined by two constraints: The maximum point-to-chord distance is less than 1/3 of the chord length, where the point-to-chord distance is computed by the following short algorithm: Draw a chord between the two end points of the trajectory and represent this chord line as $ax + bx + c = 0$, then compute the distance from a point (x_0, y_0) to the chord as

$$\frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}.$$

The second constraint on a line stroke is that the length ratio of trajectory to chord is less than 3/2. After the stroke is classified as being straight, it is assigned a stroke shape pattern according to the five orientation categories illustrated in Figure 8. Note that there are two patterns for vertical strokes. We discriminate them according to the writing order. If a vertical stroke is the final stroke of a character, the left type (Type 8 in Figure 2) is applied; otherwise the right one (Type 9 in Figure 2) is applied.

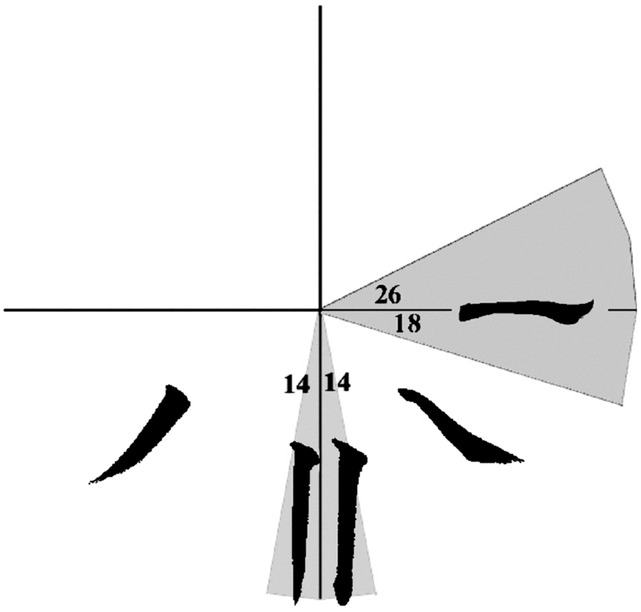

Figure 8. Orientations of straight strokes.

Two-line strokes are defined by two characteristics: the trajectory does not cross the chord, and the maximum point-to-chord distance is greater than 1/3 of the chord length. Two-line strokes comprise convex strokes and concave strokes, as shown in Figure 9, which differ in the side the trajectory occupies compared to the chord.

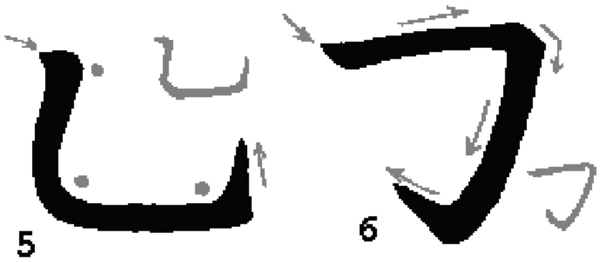

Figure 9. Examples of two-line strokes.

Multi-line strokes are composed of more than two straight lines that generally cross the chord. Currently, we do not assign any stroke shape patterns to multi-line strokes and their stroke type is considered as unrecognized.

For all unrecognized strokes, a default stroke shape pattern is applied. The default pattern can be generated by constant line mode or proportional mode, or be a pre-specified stroke shape pattern in the library. In Figure 10, we show some generated images with recognized strokes and unrecognized strokes. The black strokes are recognized and assigned corresponding shape patterns, while the gray strokes are unrecognized and drawn in proportional mode. Even though many strokes are unrecognized, we can see that the generated character images still look natural.

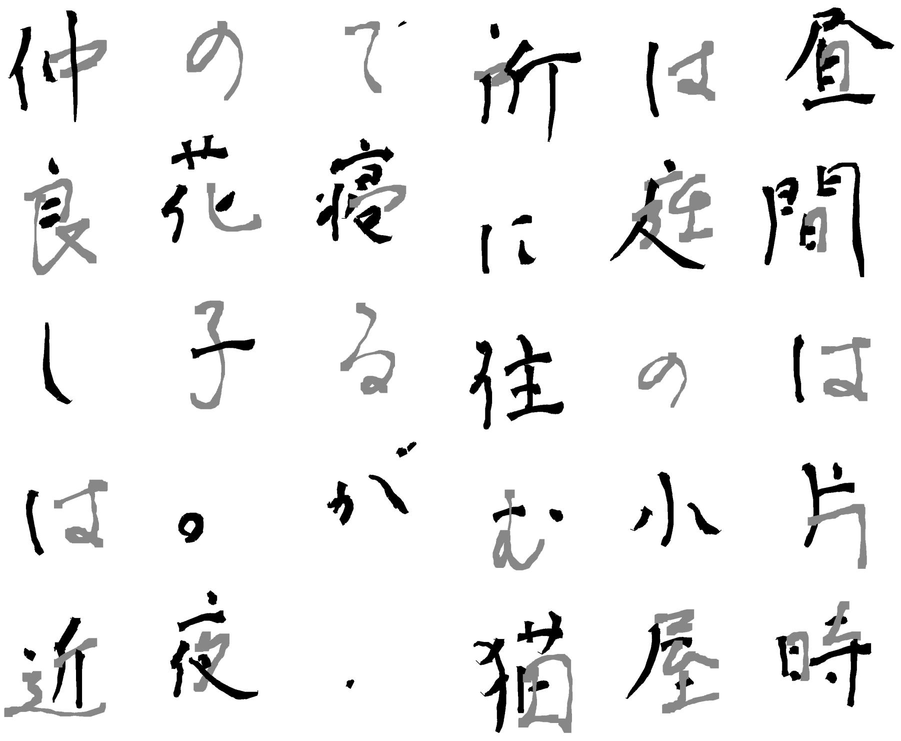

Figure 10. Combinations of recognized strokes and unknown strokes.

3.3. Calligraphic painting mode based on stroke component classification (SCC)

Our previous method PSI is based on recognition of stroke types. A decomposition of off-line Kanji characters into strokes is natural since strokes are the basic components of Japanese and Chinese characters. The stroke type of a neatly written character can usually be recognized successfully, but when the character is written quickly or in some non-elementary calligraphic style, strokes are often connected and thus cannot be classified.

In our improved SCC method, we solve this problem by dividing strokes into several parts. Each part is classified into one of three classes: "end parts", "bend parts", and "connecting parts" according to the following definitions:

(a) Each stroke has just two end parts: head and tail.
(b) The parts with high curvature are called bend parts.
(c) Bend and/or end parts are connected to each other by so-called connecting parts.

The parts defining stroke-shape are mainly end parts and bend parts because writing direction, pressure, or pen posture often change significantly in these parts. We first detect the end parts and bend parts of each stroke, and then we classify them into appropriate subclasses and draw them using a pattern from the stroke-shape library. A drawing method is the same like for previous painting mode PSI (subsection 3.2.2).

3.3.1. Detection of bending points

To use local characteristics for detecting bending points, such as the first and second order derivatives, is inappropriate because strokes are not continuous functions and are often distorted by noise. However, we can find bend parts of strokes in a way similar to detecting points with the maximum distance to a chord. The following algorithm for detecting the bending points of a stroke is robust against local noise and is applied in several variations in many on-line recognition systems [14], [15].

(a) Initialize: A = head of the input stroke; B = tail of the input stroke.
(b) Find the point C that has the maximum distance d(C,AB) to the chord AB.
(c) If d(C,AB) is higher than a given threshold, add point C to the set of bending points and find other bending points for chords AC and CB recursively {goto (b)}.

Figure 11 shows several trajectories together with the points captured by the algorithm. The bending points detected by the algorithm described above are printed boldly.

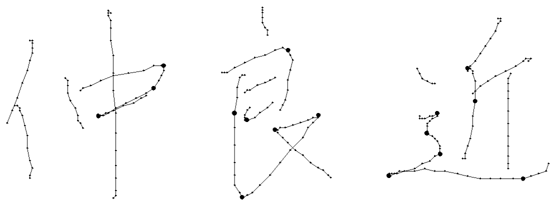

Figure 11. Detection of bending points.

3.3.2. Classification of bending points

We define four types of stroke bends as shown in Figure 12. There are three corner bends S_a, S_b, S_c and one round bend R_{abc}. We do not need to define more than one type of round bend because stroke width is approximately the same for wide and narrow round bends when being written with a brush.

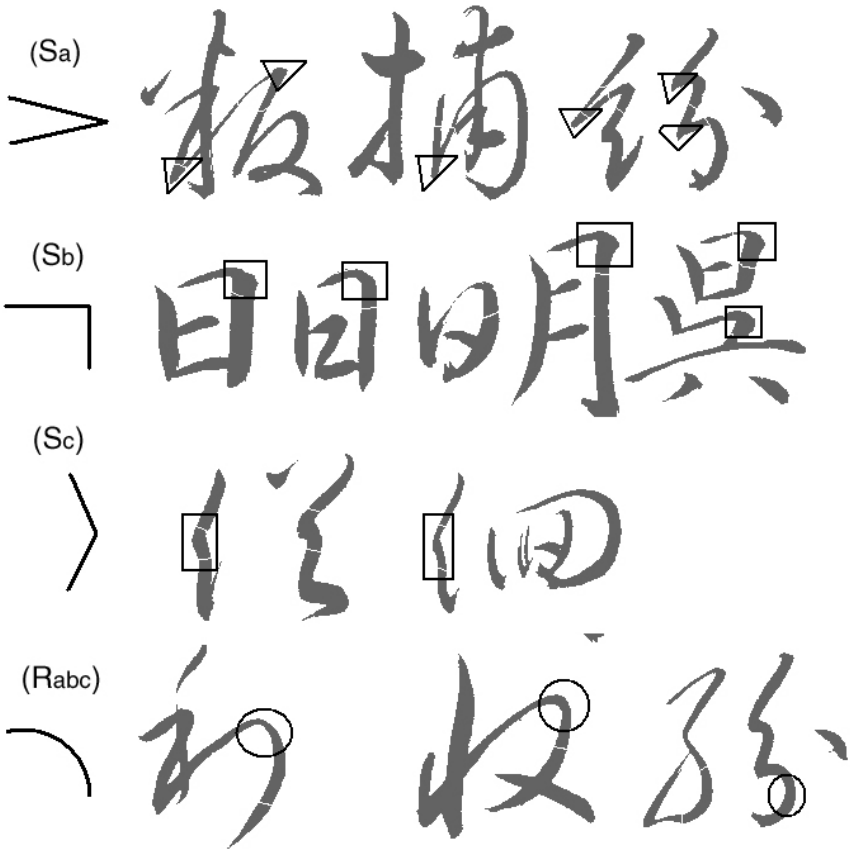

Figure 12. Examples of corner bends (S_a) (S_b), (S_c) and round bends (R_{abc}).

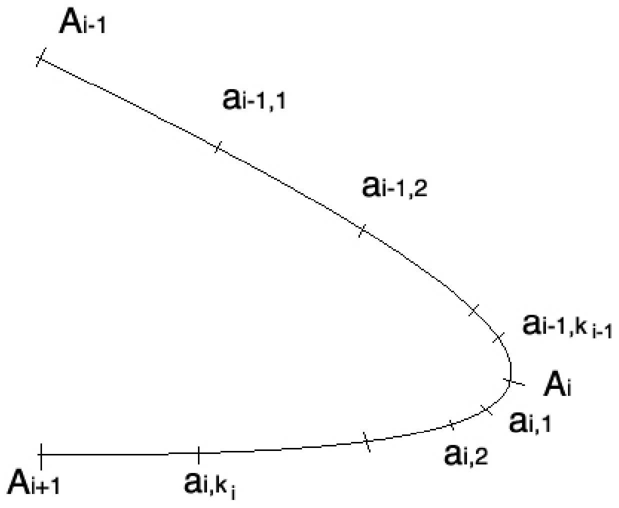

Figure 13. Stroke with three bending points: A_{i-1} ; A_i ; A_{i+1}.

These four types are sufficient to describe the most important differences of stroke width in bend parts.

Let each stroke be a sequence of n sampling points: stroke = $\{a_0; a_1;...; a_n\}$. Let us rename the indices of this sequence as follows: stroke = $\{A_0; a_{0,1};\ldots; a_{0,k0}; A_1; a_{1,1};\ldots; a_{1,k1}; \ldots ... A_{n-1,}; a_{n-1,1};\ldots; a_{n-1,kn-1}; A_n\}$, where $a_{x,y}$, $y \neq 0$, denote sampling points and $A_x = a_{x,0}$; $x\in 0...n$; are sampling points detected as bending points or end points (Figure 13). Then each bending point is classified according to the following three features (one global and two local features):

(a) Global angle α: An angle subtended by neighboring bending points: $\alpha = (A_{i-1})(A_i)(A_{i+1})$, see Figure 14-α. This global feature distinguishes three types of round bends (S_a) (S_b), (S_c).

(b) Local angle β: An angle subtended by neighboring sample points $(a_{i-1,k})(A_i)(a_{i,1})$, as shown in Figure 14-β. This local feature distinguishes the corner bends (S_a) and (S_b) from a round bend (R_{abc}). However, it is highly sensitive to noise because the definition of angle β is based only on three sampling points. Moreover, this feature cannot distinguish a corner bend (S_c) from a round bend (R_{abc}). An example follows in the next paragraph.

(c) Difference of local angles ω: The change of stroke direction in bending points, in relation to the change of direction in two neighbouring bending points. $\omega = \dfrac{(a_{i-1,k_{i-1}-1})(a_{i-1,k_{i-1}})(A_i) + (A_i)(a_{i,1})(a_{i,2})}{2} - (a_{i-1,k_{i-1}})(A_i)(a_{i,1}) = (\beta_{i-1}/2+\beta_{i+1}/2)-\beta$, look at Figure 14-ω. If a bend is smooth; i.e., Type R_{abc} in

Figure 13, then ω converges to zero, since for the round bend $\beta = \beta_{i-1} = \beta_{i+1} \Rightarrow \omega = 0°$. For a corner bend the value of the angle ω grows: $\omega \cong 180-\beta$. An example of the importance of feature (c) is given in Figure 14.ω: Feature (b) is the same for the smooth stroke sequence AXYZĀ and for the sharp sequence OXYZŌ: $\beta = 120°$, but feature (c) differs for both sequences. For OXYZŌ, ω is about 180°–120° = 60°, and for AXYZĀ, ω is about 0°.

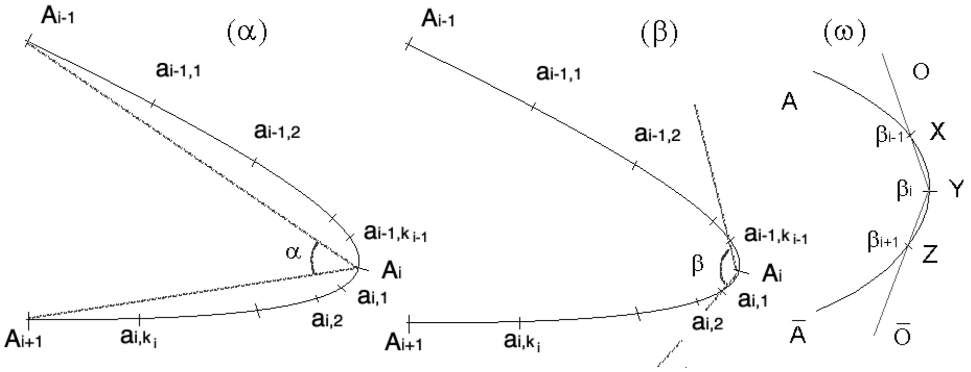

Figure 14. Graphical description of features (α), (β), (ω).

The recognition scheme that uses these three features for classifying bending points into four classes is presented in Figure 16. An example of classification based on the feature vector v (α, β, ω) is shown in Figure 15: all bending points of Figure 11 have been classified to the subclasses defined in Figure 12.

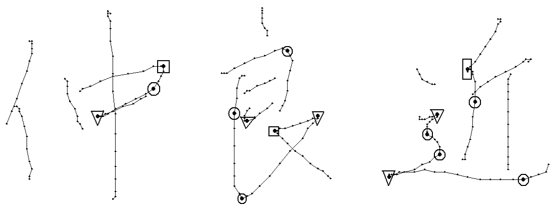

Figure 15. Classification of bending points into the subclasses defined in Figure 12.

Patterns added to the stroke-shape library (Section 3.2.1, Figure 2, Figure 5) and used for bend parts are shown in Figure 17 (original patterns) and Figure 18 (normalized patterns).

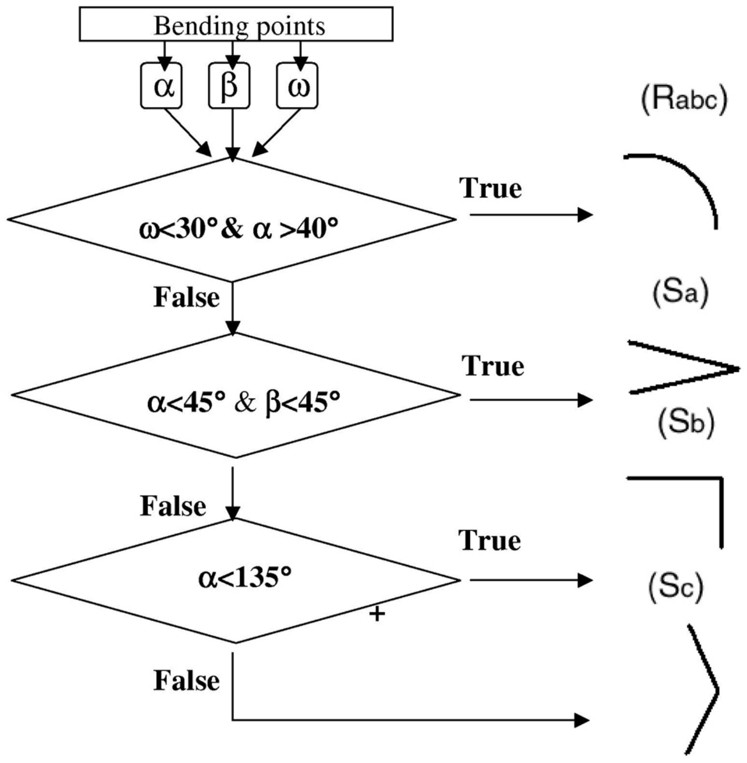

Figure 16. Recognition of bending points.

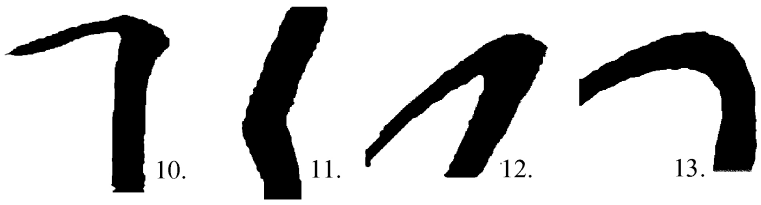

Figure 17. Stroke image templates of bend parts.

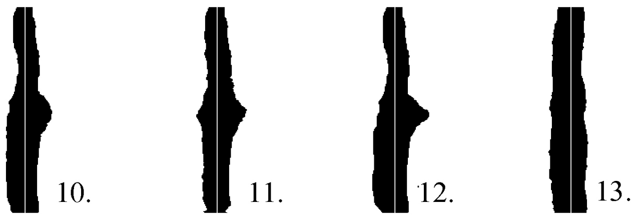

Figure 18. Normalized stroke images from Figure 17.

3.3.3. End points

Detection of end points is straightforward because they correspond to the heads and tails of strokes. We only need to recognize their types and to write them using appropriate template-bitmaps from our stroke-shape library.

For classification of end points we use a slightly modified version of our algorithm for classifying entire strokes (Section 3.2.3). The input to this algorithm, which is based on direction vectors, is the part from the end point to the nearest bending point of a stroke. To draw the end part of strokes, we use either the first half of a recognized stroke type for drawing the heads, or the second half for the tails (Figure 19). The final stroke shape of our example visualized by SCC is shown in Figure 20.

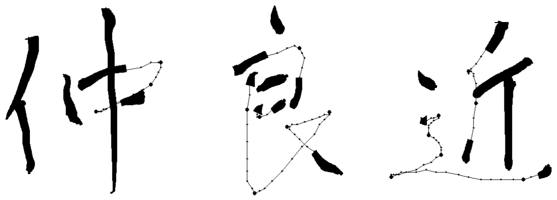

Figure 19. Recognized end parts (boldly painted).

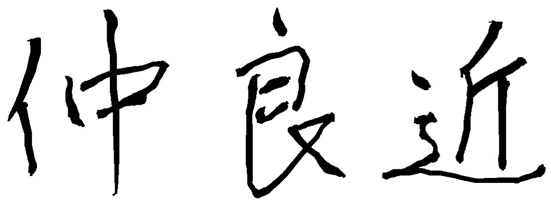

Figure 20. Strokes visualized by SCC.

4. Experiments

To validate the usefulness of our generated images, we carried out experiments for on-line and off-line character recognition using the generated images as training samples for character classifiers.

4.1. Databases used in our experiments

We use four handwritten Japanese character databases with more than 5 million samples: two off-line databases and two on-line databases. The off-line databases are ETL9B and JEITA-HP database.

The new JEITA-HP database collected in Hewlett-Packard Laboratories, Japan consists of two datasets: Dataset A (480 writers) and Dataset B (100 writers). Datasets A and B were collected under different conditions, so we will report results for each part separately. Generally speaking, Dataset B is written more neatly than Dataset A. The entire database consists of 3214 character classes (2965 kanji, 82 hiragana, 10 numerals, 157 other characters /English alphabet, katakana and symbols/). The most frequent hiragana and numerals are twice in each file. However, in this paper we use only Kanji and hiragana characters for our experiments, which form the intersection of all our databases. In [19], [20], Kawatani has introduced some experimental results on this database.

The two on-line databases: Kuchibue and Nakayosi [10]-[12] from which we generated our samples, were already introduced in Section 2. We generated from each on-line pattern five off-line images, always using different painting modes. The size of the generated images is 96 × 96 pixels. In Table 1, we compare the characteristics of all databases. From these databases we selected patterns for training and testing. A specification of some of these test files is given in Table 2.

Table 2. Specification of training and testing files.

test/train files	Source	#Category	#Pattern		Comments
			Per category	Total	
test-Etl	ETL9B	3036	100	1,435,440	even samples
train-Etl	ETL9B	3036	100	1,435,440	odd samples
test-Hp_A	JEITA-HP A	3036	66	607,200	last 66 writers
test-Hp_B	JEITA-HP B	3036	14	60,720	last 14 writers
test-Mdb	Kuchibue	3036	600	1,917,480	5methods × 120 original
train-Nky	Nakayosi	3036	800	2,428,800	5methods × 160 original

4.2. Classifiers used in our experiments

We use two off-line recognition schemes in our experiments. The first recognizer represents each character as a 256-dimensional feature vector. It scales every input pattern to a 64 × 64 grid by non-linear normalization and smoothes it by a connectivity-preserving procedure. Then, it decomposes the normalized image into 4 contour sub-patterns, one for each main orientation. Finally, it extracts a 64-dimensional feature vector for each contour pattern from their convolution with a blurring mask (Gaussian filter).

For our first classifier (MQDF) a pre-classification step precedes the actual final recognition. Pre-classification selects the 100 candidate categories with the shortest Euclidian distances between their mean vectors and the test pattern. The final classification uses a modified quadratic discriminant function (MQDF2) developed by Kimura [16] from traditional QDF. Our second classifier (LVQ) uses only 196-dimensional feature vectors; i.e., 49 features for each orientation. It is based on the MCE training method (minimum classification error) proposed by Juang and Katagiri [17, 18].

4.3. Generating samples for off-line recognition

4.3.1. Experiment I

In this experiment we investigated whether generating more complex (highly realistic) patterns is more beneficial to off-line recognition performance, if a trained recognizer is tested on real off-line images.

We generated four training sets from a subset of the Nakayosi on-line database [10], (using constant mode, proportional mode, PSI, and SCC). Each set contains 150 samples for 3036 categories (2965 Kanji, 71 Hiragana). The image size of the generated patterns is 96 × 96 pixels. These four training sets are generated from the same on-line patterns; the difference is only in the generating method. The fifth training set contains real off-line patterns.

We tested the MQDF classifier trained with five different training sets on two real off-line test. Our achieved recognition rates are shown in Table 3.

Table 3. Off-line recognition rates of MQDF trained with generated patterns and real off-line data.

Train	Painting mode for on-line data				off-line
	constant	proportion	PSI	SCC	---
test-Etl	91.82	94.68	95.75	95.97	97.75
test-Hp	88.78	92.63	93.61	93.89	95.53

We see in this experiment that PSI and SCC outperform simple constant and proportional painting modes. Although none of our generated image sets beat real off-line training dataset, the difference is acceptable. The results of PSI and SCI are closer to the recognition rates of real off-line training data than the rates of proportional and constant modes. We suppose that they generate more realistic samples, which are more similar to real characters.

4.3.2. Experiment II

In this experiment, we investigate whether adding generated patterns to real patterns can enrich training sets and improve recognition rates. We use the 100 samples with odd index per category from ETL9B database as our basic training set. Then we add another 700 samples generated by PSI, SCC, Proportional painting mode and constant mode thickness 3 and 5 from 160 original on-line samples of Nakayosi database (only 700 from 800 generated samples are used). We train the two classifiers (MQDF, LVQ) using the first 50, 100, 200, 400 or all 800 samples from the training set. In Table 4 and Table 5 we see that the contribution of our generated patterns to the improvement of recognition rate is 1.26%–1.35% for MQDF classifier and 1.96%–2.26% for LVQ classifier. The greater improvement is for fluently written characters from JEITA-HP dataset A than for neat samples from JEITA-HP dataset B. This is because our on-line databases Kuchibue and Nakayosi were mostly written fluently. The results for pattern generated from Kuchibue database are treated separately, because it is not an original off-line database.

Table 4. Test results of LVQ classifier trained with 50-800 samples per category.

LVQ	50	100	200	400	800	
HP-A	85.62	87.40	88.75	89.35	89.66	+2.26
HP-B	90.87	92.19	93.33	93.65	94.15	+1.96
MDB	80.84	83.15	87.46	88.51	89.62	+6.47

Table 5. Test results of MQDF classifier trained with 50-800 samples per category.

MQDF	50	100	200	400	800	
HP-A	88.81	90.61	91.65	91.74	91.96	+1.35
HP-B	93.61	95.12	95.93	96.24	96.38	+1.26
MDB	85.00	87.94	91.47	91.89	92.88	+4.94

4.4. Off-line methods for on-line recognition

This experiment is to test the eligibility of painting modes for using off-line method for recognizing on-line patterns. We divided each of four training sets (from experiment 4.3) into two subsets: a training set (100 samples per category) and a test set (50 samples per category) so that training and test data were generated by the same method. In Table 6 we see that the recognition rate is decreasing for sophisticated methods PSI, SCC, which generate more realistic patterns. This is not surprising because a good feature function should be invariant to deformations in stroke shape. To extract feature information from a stroke of constant width is simpler than from a more complex shape. So our result from this experiment is that in case an off-line approach is used for on-line patterns, bitmaps should be generated by the simplest method, or the feature vector should be generated directly from the vector representation.

Table 6. On-line recognition rates with off-line method.

Mode	constant	proportional	PSI	SCC
Rec. rate	95.25	95.12	94.36	93.50

Generally speaking, using off-line methods for on-line recognition cannot outperform genuine on-line methods. Nevertheless, in Reference [22] we present experiments, where an off-line approach yields better recognition rates. In this reference, both on-line and off-line recognition are tested on on-line data written by elderly people with an average age of 70.5 years. Those patterns are casually written, very often with an untypical stroke order, which is a considerable problem for on-line recognizers.

5. Conclusion

We demonstrated two calligraphic painting modes for generating highly realistic images. The uniqueness of this approach is that we don’t guess stroke shape from on-line information, but apply a stroke shape of real patterns. In experiments we showed that the generated patterns enrich the training dataset for off-line recognizers so as to achieve higher recognition performance. Although these methods cannot create as many pattern variations as deformation techniques can, our method creates more realistic patterns and, if needed, it can be combined with deformation techniques before generating stroke shape. Our method allows generating patterns that strongly resemble genuine off-line patterns written with a brush pen, which is useful for postal address recognition, where collecting a large number of brush-written samples is hard.

Another application is generation of dual on-line/off-line databases. This database duality is very rare, because it is very difficult to collect dual databases simultaneously, as was done for instance in [21]. Our method is suitable for generating large, realistic off-line parts of dual databases. Using the dual database is very useful for investigations of combined on-line/off-line classifiers, as shown in [22]. Also, using both on-line and off-line methods for the same on-line data can significantly improve recognition rate. For instance, in [22] the recognition rate of combined on-line and off-line classifiers is 94.14% while the best single on-line and off-line recognizer yield only 85.67% and 89.07% respectively for a Japanese Kanji test set written by elderly people.

Another field, where these realistic patterns can be useful are pen-based graphical interfaces, where these methods can well visualize characters written on a non-paper medium. Or, it can be employed for generating user-specific fonts, expressing the uniqueness of each writer.

And finally, we plan to make off-line versions of our already publicly available databases Kuchibue and Nakayosi, each contains more than three million patterns, and release these dual databases for public use, too.

References

[1] T.H. Hilderbrand and W. Liu, “Optical recognition of Chinese characters: advances since 1980”, *Pattern Recognition* 26(2), 1993, 205–225.

[2] F. Kimura, *et al.*, “Improvement of handwritten Japanese character recognition using weighted direction code histogram”, *Pattern Recognition* 30(8), 1997, 1329–1337.

[3] H. Nagahashi and M. Nagatsuyama, “A pattern description and generation method of structural characters”, *IEEE Trans. Pattern Analysis and Machine Intelligence* 8(1), 1986, 112–117.

[4] C.-H. Tung, Y.-J. Chen and H.-J. Lee, “Performance analysis of an OCR system via an artificial handwritten Chinese character generation”, *Pattern Recognition* 27(2), 1994, 221–232.

[5] V.K. Govindan and A.P. Shivaprasad, “Artificial database for character recognition research”, *Pattern Recognition Letters* 12(10), 1991, 645–648.

[6] G. Ghosh and A.P. Shivaprasad, “An analytic approach for generation of artificial hand-printed character database from given generative models”, *Pattern Recognition* 32(6), 1999, 907–920.

[7] T. Ha and H. Bunke, “Off-line handwritten numeral recognition by perturbation method”, *IEEE Trans. Pattern Analysis and Machine Intelligence* 19(5), 1997, 535–539.

[8] M. Mori, *et al.*, “Generating new samples from handwritten numerals based on point correspondence”, *Proc. 7th IWFHR*, 2000, pp. 281–290.

[9] I. Guyon, *et al.*, “UNIPEN project of on-line data exchange and recognizer benchmarks”, in *Proc. 12th ICPR*, 1994, vol. II, pp. 29–33.

[10] M. Nakagawa, *et al.*, “On-line character pattern database sampled in a sequence of sentences without any writing instructions”, in *Proc. 4th ICDAR*, 1997, pp. 376–380.

[11] S. Jaeger and M. Nakagawa, “Two on-line Japanese character databases in Unipen format”, in *Proc. 6th ICDAR*, Seattle, 2001, pp. 566–570.

[12] K. Matsumoto, T. Fukushima and M. Nakagawa, “Collection and analysis of on-line handwritten Japanese character patterns”, in *Proc. 6th ICDAR*, Seattle, 2001, pp. 496–500.

[13] H. Tanaka, *et al.*, “Hybrid pen-input character recognition system based on integration of online-offline recognition”, in *Proc. 5th ICDAR*, 1999, pp. 209–212.

[14] U. Ramer, “An interactive procedure for polygonal approximation of plane closed curves”, *Computer Graphics and Image Processing* 1, 1972, 244–256.

[15] K. Ishigaki and T. Morishita, “A top-down online handwritten character recognition method via the denotation of variation”, in *Proc. ICCPCOL*, 1988, pp. 141–145.

[16] F. Kimura, *et al.*, “Modified quadratic discriminant function and the application to Chinese characters”, *IEEE Trans. Pattern Analysis and Machine Intelligence* 9(1), 1987, 149–153.

[17] B.-H. Juang and S Katagiri, “Discriminative learning for minimization error classification”, *IEEE Trans. Signal Processing* 40(12), 1992, 761–768.

[18] C.-L. Liu and M. Nakagawa, “Evaluation of prototype learning algorithms for nearest neighbor classifier in application to handwritten character recognition”, *Pattern Recognition* 34(3), 2001, 601–615.

[19] T. Kawatani and H. Shimizu, “Handwritten Kanji Recognition with the LDA Method”, in *Proc. 14th ICPR*, Brisbane, 1998, Vol. II, pp. 1031–1035.

[20] T. Kawatani, “Handwritten Kanji recognition with determinant normalized quadratic discriminant function”, in *Proc. 15th ICPR*, 2000, Vol. 2, pp. 343–346.

[21] C. Viard-Gaudin, P.M. Lallican and P. Binter, “The IRESTE on/off (IRONOFF) dual handwriting database”, in *Proc. 5th ICDAR*, 1999, pp. 455–458.

[22] O. Velek, S. Jaeger and M. Nakagawa, “A new warping technique for normalizing likelihood of multiple classifiers and its effectiveness in combined on-line/off-line Japanese character recognition”, accepted for *8th IWFHR,* 2002.