A Method of Generating Calligraphy of Japanese Character using Deformable Contours

Lisong Wang Tsuyoshi Nakamura Minkai Wang Hirohisa Seki Hidenori Itoh
Department of Intelligence and Computer Science,
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya 466, Japan.

Abstract

This article considers the problem of generating various calligraphy of Japanese character from some limited existing fonts. We propose a generation method based on the deformable contour model *g-snake*. By representing the outline of each stroke that makes of a character with a g-snake, we cast the generation problem into global and local deformation of *g-snake* under different control parameters, meanwhile, the local deformation obeys the energy minimization principle of *regularization* technique. The base value of the control parameters are learned from given sample fonts. The experimental results, i.e. the generated calligraphy show such a process as a reasonable way of generating characteristic calligraphy.

1 Introduction

Writing style varies from person to person. Inherently, imitating and generating diverse calligraphy of a character automatically with a computer is always demanded. However, because of the shape variety of the strokes making up a character and the complexity of the shape deformation, it is usually difficult to capture the inner relationship between different writing style by simple rules. Therefore, to cope with the situation, currently two measures are mainly adopted for calligraphy generation: one is preserving some different fonts in advance, and giving out a type of writing among them opportunely, as used in almost all editor or printing tool, and the other one is using some manually defined regulations based on some experience, as applied in many calligraphy systems [Nakamura *et al.*, 1994, 1995, Zhang *et al.*, 1993.]

Obviously, the first method lacks flexibility, the number of type of calligraphy generated is font-limited, and enormous computer resources are required with the increase of amounts of the font. The second one lacks generality, it is based on trial-and-error, fine result can be obtained from it without special knowledge about calligraphy. The problem of generating different calligraphys from limited existing fonts with human intervention as little as possible is left.

More formally, our problem may be formulated as follows: Given two sample calligraphys C_0, C_1 and $t \in [0,1]$, construct versatile intermediate object C_t together with the increasing of t, where, C_t is a more violent deformation in resembling tendency from C_0 to C_1. In a way, this is a multidimensional shape interpolation problem, and, as such, is ill-posed, in the sense that there are many possible interpolants C_t satisfying this very vague condition.

There are also some research carryed out for the shape interpolation problem [Sederberg *et* al., 1992]. As a common consent, the *regularization* [Poggio *et* al., 1985] technique is a powful tool of transforming a ill-posed problem into a well-posed one. By finding some natural conditions which are not explicit in the input it reduces the number of solutions. In this paper, we propose a method for dealing with the problem of generating calligraphy from limited fonts by applying *regularization* technique.

We apply *regularization* to our calligraphy generation problem grounded on following argument;

> A different calligraphy of a character is a deformation of its stroke contours satisfying energy minimum principle under some external constraints.

Some deformable contour models based on the energy minimum principle of *regularization* have been developed and successfully applied to many aspects of machine vision. Kass first established a well-known active contour model, the *snake* model to process low-level vision tasks such as edge detection, boundary formulation and stereo and motion matching [Kass *et* al., 1988]. To achieve the aim of contour extraction of arbitrary object under noisy image [Lai and Chin, 1995], Lai gives a more innovative one, so-called generalized active contour, the *g-snake* model, based on the stochastic relaxation theory [Geman, 1995]. The common feature of both models is that, for each model, there is a defined internal energy, which is used to keep the initial shape of a contour, and an imaginary external force acted on the contour. The contour is attracted by such actions to a desirable final state, where the sum of internal and external energy of the contour is minimal. Due to the different usage, the external force could be image intensity, edge, termination etc. An advantage of g-snake is that it is capable of representing any arbitrary shape, not only accounts for global changes due to rigid motions, but also retains the ability for local control.

We argue that to deal with the calligraphy generation problem, we must consider both global and local deformation of a stroke. Accordingly, the g-snake is a fit table contour model for our task. We develop a calligraphy generation method based on g-snake. In our method, we use a pair of existing font data as samples, one is for C_0, another is for C_1, represent each stroke of a sample by a closed g-snake, investigate global deformation between corresponding strokes, imagine that there are pulling forces acted on C_0 to make it deform to $C\$, and learn the contour deformable control parameter by using the $C\$ as the final convergence of energy minimization from initial position Co. Then we can control the generation of a series of interpolation C_t by varying the learned parameters.

The remaining of this article is arranged as follows: In section 2, we introduce g-snake model, in section 3, we describe the application of g-snake model for our problem in detail, in section 4 we show some examples of experimental results and discuss about them, and finally, the conclusion and the future work is summarized in section 5.

2 Deformable Contours Model: g-snake

In [Lai and Chin, 1995], the major role of the g-snake model is to deal with an ill-posed problem, extracting contour of any object from an observational image.

2.1 Representing Contour of an Object

The g-snake model represents a contour as a link of point vector, $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n]$, where, each $\mathbf{u} \in E = \{(x,y) : x, y = 1, 2, \cdots, M\}$, thus $\mathbf{U} \in E^n$. Such a point is also called snaxel (snake pixel). Each snaxel $\mathbf{u}_i$ can be expressed as a linear combination of its two adjacent snaxel vectors:

$$\mathbf{u}_i = \alpha_i \mathbf{u}_{i_\alpha} + \beta_i \mathbf{u}_{i_\beta} \tag{1}$$

where the basis indices are given by:

$$i_\alpha = \begin{cases} i-1; & i > 1 \\ 3 & i = 1 \end{cases} \quad i_\beta = \begin{cases} i+1; & i < n \\ n-2 & i = n \end{cases} \tag{2}$$

Accordingly, the shape equation of the contour of an object can be written down as:

$$\mathbf{A}\mathbf{U}^T = 0 \tag{3}$$

where $\mathbf{A}$ is called shape matrix that contains the necessary information to describe the shape:

$$A = \begin{bmatrix} 1 & -\beta_1 & -\alpha_1 & 0 & \cdots & 0 \\ -\alpha_2 & 1 & -\beta_2 & 0 & 0 & \cdots \\ 0 & -\alpha_3 & 1 & -\beta_3 & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & -\alpha_{n-1} & 1 & -\beta_{n-1} \\ 0 & 0 & \cdots & -\beta_n & -\alpha_n & 1 \end{bmatrix}$$

A contour $\mathbf{U}$ is called *nontrivial* if $|x_{i_\alpha} y_{i_\beta} - x_{i_\beta} y_{i_\alpha}| > 0$ for $1 \leq i \leq n$. Such shape representation is account for both global and local deformations:

- **Global Deformations**

  Global deformations correspond to rigid motion of contour such as scaling, rotation, stretching and dilation. It has been proven in [Lai and Chin, 1995] that the contour representation mentioned above keeps the following invariance property for a contour:

  Two nontrivial contours satisfy the same shape equation if and only if they are related by a linear transformation.

  Therefore, according to this invariance property, the global deformation of a contour can be computed simplely by affine transformations without any influence on the shape matrix.

- **Local Deformations**

  The g-snake models the local deformations as random fluctuation of snaxels in a interested family of contours $\mathbf{U} \in \Omega \supseteq E^n$. An internal energy induced by shape matrix $\mathbf{A}$ is defined to represent such fluctuation:

$$E_{int}(\mathbf{U}) = \frac{(\mathbf{A}\mathbf{U}^T)^T \mathbf{R}^{-1} (\mathbf{A}\mathbf{U}^T)}{l(\mathbf{U})} \tag{4}$$

  where $\mathbf{R} = diag\{\sigma_1^2, \sigma_2^2, \cdots, \sigma_n^2\}$ are the deformation variances σ_i^2 of snaxel i, $l(\mathbf{U})$ is a normalizing constant:

$$l(\mathbf{U}) = \frac{1}{n} \sum_{i=1}^{n} \| \mathbf{u}_{i+1} - \mathbf{u}_i \|^2$$

  Now the probabilities of contour $\mathbf{U}$ may be assigned as expression 5 which is so-called *Gibbs measure*:

$$p(\mathbf{U}) = \frac{1}{Z} exp(-E_{int}(\mathbf{U})) \tag{5}$$

  where Z is also a normalizing constant:

$$Z = \sum_{U \in \Omega} exp(-E_{int}(\mathbf{U}))$$

Equivalently, expression 5 also defines the conditional probability of *Markov random field* to yield prior distributions for any arbitrary contour.

$$p(\mathbf{u}_i|\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n) = p(\mathbf{u}_i|\mathbf{u}_{i_\alpha}, \mathbf{u}_{i_\beta}) \quad (6)$$

The expression 6 means that a contour $\mathbf{U}$ is entirely specified given all probability of $\mathbf{u}_i$, while a probability of $\mathbf{u}_i$ is specified given its two adjacent basis snaxels.

2.2 Maximum Posterior Estimation

The g-snake model treats the contour extraction task as a process to estimate unknown deformation of contour from an image. By using Bayesian framework, it can be realized as maximum posterior(MAP) estimation. Rewriting and denoting the internal and external energy as follows:

$$E_{int}(\mathbf{u}_i) = \frac{\| \mathbf{u}_i - \alpha_i \mathbf{u}_{i_\alpha} - \beta_i \mathbf{u}_{i_\beta} \|^2}{l(\mathbf{U})} \quad (7)$$

$$E_{ext}(\mathbf{u}_i, \mathbf{g}) = 1 - \mathbf{h}_i^T \mathbf{f}(\mathbf{u}_i + \mathbf{g}) \quad (8)$$

where $\mathbf{g}$ is an arbitrary reference point, $\mathbf{h}_i$ is an unit vector indicates the interested deformation direction of a snaxel on the contour, and function $\mathbf{f}(x)$ corresponds the external force, which can be edge intensity for example, depends on task that pulls the contour to desirable position.

Then contour extraction task turns into solving problem of MAP estimation, formulated as an energy minimization problem of finding $\mathbf{U}_{map}$ and $\mathbf{g}_{map}$ in expression 9.

$$\{\mathbf{U}_{map}, \mathbf{g}_{map}\} = \arg\min_{\mathbf{U},\mathbf{g}} \sum_{i=1}^{n} \left\{ \frac{\lambda_i}{1-\lambda_i} E_{int}(\mathbf{u}_i) + E_{ext}(\mathbf{u}_i, \mathbf{g}) \right\} \quad (9)$$

where

$$\lambda_i = \frac{\sigma_\eta^2}{\sigma_\eta^2 + \sigma_i^2} \in [0, 1]$$

are the local *regularization* parameters which control the amount of local template deformation. σ_η^2 is the Gaussian distribution of the noise contained in force measurement.

3 Calligraphy Generation System

The different calligraphy of a character is indeed the deformation of its strokes in a lot of styles. Because of the deformation complexity, it is difficult to arrange simple and plain rules to comprehend deformations of every points make up a stroke. Our basic thinking is to investigate the deformation of several salient feature points on sample calligraphy C_0 and C_1, assuming the deformation is owing to some forces coming from C_1 and acting on the outline of strokes of C_0, and it will convergence to somewhere where the sum of energy of points is minimum so that we can charge the position determination of remaining points with the energy minimum processing, and hence, to generate versatile calligraphy the deformation of strokes can be simply described as the energy minimum principle. Up to now, we use only Japanese alphabet, the Hiragana, for the calligraphy generation. So we describe our system concentrated on processing of Hiragana.

3.1 Processing Flow

Given two samples *C_0,C_1,* in order to generate varying calligraphys C_t, our system proceeds in following stages:

- Pre-process.
- Extract strokes of from input characters.
- Complete the correspondence completion between each stroke of C_0 and *C\*.
- Initialize a g-snake on each stroke.
- Investigate global deformation between corresponding strokes.
- Learn g-snake parameter to control local deformation.
- Varying the learned parameters to generate new calligraphy.

3.2 Pre-processing

Giving two sample calligraphys Co,C_1 as the input, because they are all binary images, By investigating whether the 8 neighbors of a pixel are all black, we can easily get binary array matrix T_0,T_1, whose elements corresponds to each pixels of C_0, *C_1*, take value of 1 if a boundary pixel or 0 for a non-boundary pixel.

3.3 Stroke Extraction

Since the stroke is the basic processing unit in our system, we must decompose sample calligraphy to separate stroke set. In contrast to no extra work just an boundary tracing needs for a Hiragana without intersection, procedure for a intersection one is slight tiresome.

At this time, the processing is semi-automatic, i.e. the salient feature points must be decided manually. Having *n* feature points clicked near the related stroke of each sample Co and *C\*, a stroke is divided into *n* segments, then the processing saves their coordinates in two lists $\bar{K_0} = [k_{00}, k_{01}, \cdots, k_{0n}]$ and $\bar{K_1} = [k_{10}, k_{11}, \cdots, k_{1n}]$ at

Figure 1: Salient Feature Clicked

first, where, the elements in K_0 and K_1 with the same subscript are thought as correspondence. We call it a partial correspondence. For example, to extract the first stroke of Hiragana "a", points dotted in Figure 1 have to be clicked.

Following processing is: aligning each points clicked i.e. all elements in $K_l(l \in 0,1)$ on their nearest boundary by matching it to $T_l(l \in 0,1)$, obtained in pre-processing stage. For each pair $k_{bi}, k_{b(i+1)}, 0 \leq i < n, b = 0$ *or* 1, using $T_l(l \in 0,1)$ to trace and record a route along the boundary of the stroke(for the point last clicked searching a route to the first clicked one), If an intersection is encountered, the program will switch boundary tracing operation to spline interpolation, to fill the interval of boundary due to the intersection. Figure 2 is an example of decomposing Hiragana "a" and "u" into strokes.

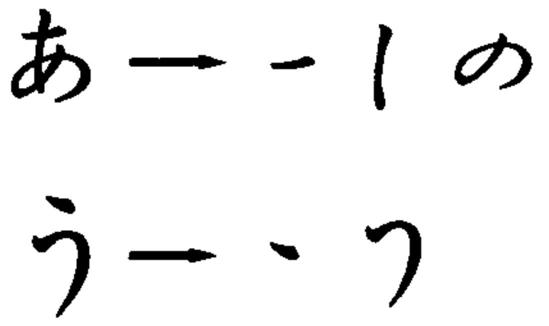
Figure 2: Decomposing calligraphy into strokes

3.4 Correspondence Completion

In this stage, the partial correspondence of feature points will be turned into a full correspondence of snaxels. Candidate snaxels to initialize a g-snake are selected from each route between two feature points produced at last-stage for each related stroke, and an full correspondence for related stroke of two samples is established. Given a space between the snaxel, the necessary number and coordinates of snaxels for a element in Co to its next one can be calculated except the last one who's coming element is the first element. Then, for an element in C\, assuming the number of snaxel to next element should be the same as its correspondence element in Co, the proper adjusted space can be calculated and snaxel can be selected from the route produced for C\ in last stage. Finally, a full correspondence is obtained by binding these selected snaxels orderly.

3.5 G-snake Initialization

To prepare for the deformation, using the snaxels found in the last stage, a closed g-snake is located on the contour of stroke, which contains the necessary shape infor-

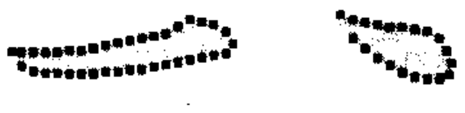
Figure 3: Locating g-snake on a stroke

mation. Figure 3 is the example of locating two g-snakes on the first stroke of Hiragana "a" and "u":

3.6 Investigate Global Deformation

In this processing stage, the global deformation of rigid motions such as rotation scale and stretching and dilation arose between sample stroke is investigated. Because the rotation and the stretching have the strongest visual appeal, we investigate rotation and stretching between each related stroke of sample C_0 and C_1. The investigation occurs along the g-snakes in the unit of a pair of partial correspondence segments. Supposing there are m segments on one g-snake, we first put the start snaxel on C_1 together with that on C_0 by translation, then by comparing their coordinates of start and end snaxel, we can know the necessary stretching and rotation angle for C_0 to deform to C_1, and put them in list $S_U = (\vec{s}_1, \vec{s}_2, \cdots, \vec{s}_m)$ and $\Theta_U = (\theta_1, \theta_2, \cdots, \theta_m)$. For example shown in Figure 4, assuming arc AB and AB' are contour correspondence segments, the global stretching is calculated by comparing the length of line segment AB and AB', and the rotation angle α is the angle between them.

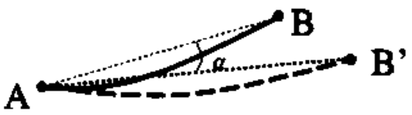

Figure 4: Global deformation of stroke

3.7 G-snake control parameter learning

In our system, the local deformation is simulated as the energy minimization process of closed g-snake on the outline of stroke. The function f(u,) describing the external energy in expression 8 is designated as a pulling force acted on snaxel on Co, with direction points to its correspondence snaxel on C\ and the strength in proportion to the normalizing constant /(U) is described in the last section. Therefore, using the correspondence table created in previous stage and putting the heads of g-snake on C\ together with its corresponding global deformed one on Co, such directions and strengthes to form a force map is easily generated. Then we use this force map to minimize the total energy for the g-snake, and the local regularization parameters λ_i in expression 9 can be estimated by *local minimax criterion[Lai* and Chin, 1993].

3.8 Generate new Calligraphy

Having known global deformation parameter S_{U}, Θ_{U} and local regularization parameter λ_i, we can generate a new calligraphy by giving a different control variable $t \in (0,1)$, deform each stroke segment globally using affine transformation under parameters tS_{U}, $t\Theta_{U}$, and then deform it locally by g-snake energy minimization with regulation parameter $t\lambda_i$. This control the deformation from weakly to violently.

4 Experiment Results

This section shows some examples of generated calligraphy. In order to test the robust of our algorithm, we deliberately selected four kinds of Calligraphy fonts of Japanese "Hiragana" existing in our computer system which are called *kaisho, gyosho, mini and gothic.* These fonts have clearly different appearance. Calligraphy of such fonts for Hiragana "a" are shown in Figure 5:

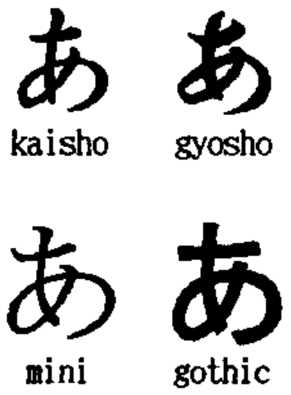

Figure 5: Different type of calligraphy

Using a pair of calligraphy fonts of Hiragana "a" as samples, and executing the algorithm described in last-section, some generated examples are shown in Figure 6.

In Figure 6, calligraphys of the first row is used as sample C_0 and last row for *C*. This is done to test whether our algorithm is available for sample of any selected type. Therefore, the first column in Figure 6 is an progressively proceeding from *kaisho* to *gyosho,* the second column is *kaisho* to *mini,* and the third is *gothic* to mini. After learning global and local deformation parameters, the calligraphys in second and third row are generated by setting *t* to 0.3 and 0.6.

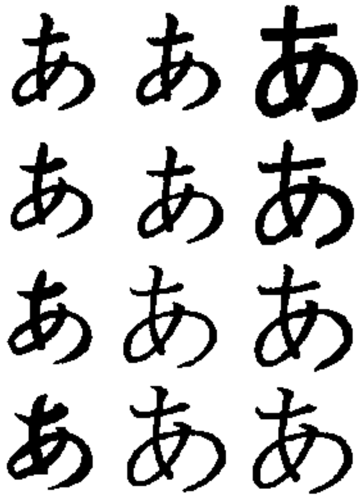

Figure 6: Generated calligraphy

Figure 7 gives generated calligraphys for some other Hiragana "i", "u" "e", and "o"of Japanese, the selected generation condition is: from the first row to the fourth row, the control parameter is set to be 0.1, 0.3, 0.6 and 0.9 respectively. We can observe from this picture that the increase of t makes the lose the peculiarity of Co, simultaneously, get that of *C*.

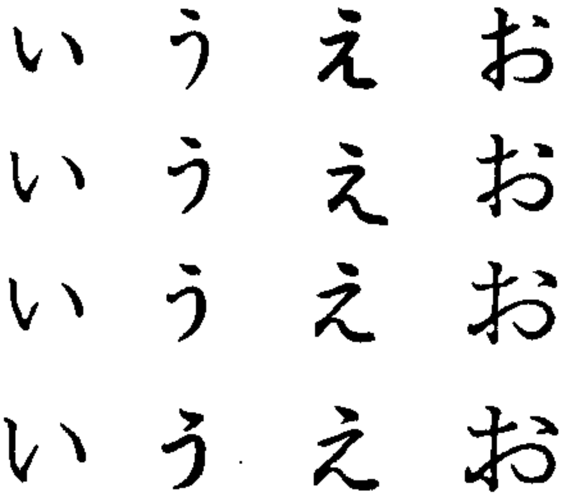

Figure 7: Other generated calligraphy

We also test our method on processing real calligraphy. The right down of Figure 8 is a real calligraphy of Hiragana "a" read from a scanner. By using it and existing Hiragana "a" of font "kaisho" (left up of Figure 8) as samples, we can generate intermediate calligraphy,

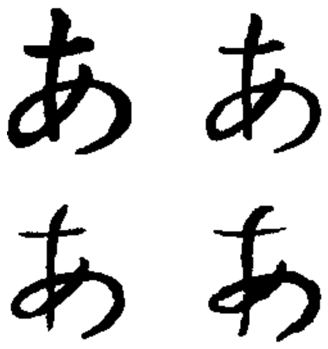

Figure 8: Generated calligraphy from real sample

the right up and the left down shown in Figure 8 with *t* selected as 0.3 and 0.8.

4.1 Discussion

We can know from these figures, that our algorithm do generate the medium states of samples, and the sample calligraphy can be any type of font including the real calligraphy. The deformation can be controlled by varying the parameter *t*. The results are similar to calligraphy written by real people. But we have to indicate that something must be improved is also left over. Some generated results are not acceptable in an aristic point of view. With the increase of *t,* the character generated looses the rhythm of the initial character, or it is a non canonical way. Because the final position of snaxel is decided with the energy minimum and due to the control difficulty of this processing, some strokes of the generated character are locally blurred. We want to overcome these shortcomings in our next system version.

5 Conclusion

We demonstrate the problem of generating various calligraphy of Japanese character from different fonts can be modeled as global and local deformation of its stroke contour obeying the energy minimization principle of *regularization* technique. Therefore, we apply the deformable contour model g-snake solving the calligraphy generation problem. Such a method makes the generation more flexible and the generated results are more natural. Some useful left work can be indicated as: extracting strokes of a character automatically, modeling the force or pressure applied on the brush and the dynamic of writing kana or kanji, interpolating writing scratch between strokes, i.e the semicursive style of writing, and the extension of the existing system to calligraph processing of Chinese characters.

References

[Geman, 1995] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. PAMI-6, pages 721-741, 1984.

[Kass *et al,* 1988] M. Kass, A. Witkin and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision,* pages 321-331, 1988.

[Lai and Chin, 1993] Kok F. Lai and Roland T. Chin. On regularization, formulation, and initialization of the active contour models (snakes). *Asian Conference on Computer Vision,* pages 542-545, 1993.

[Lai and Chin, 1995] Kok F. Lai and Roland T. Chin. Deformable Contours: Modeling and Extraction. *IEEE Transactions on pattern analysis and machine intelligence,* Vol. 17, No. 11, November, pages 1084-1090, 1995.

[Nakamura *et al,* 1994] T. Nakamura, H. Seki and H. Itoh. A Calligraphy System based on Analyzing User Writing Speed. *Proceeding of The 3rd International Conference on Fuzzy Logic, Neural Nets and Soft Computing,* pages 189-190, 1994.

[Nakamura *et al,* 1995] T. Nakamura, M. Yamada, H. Seki and H. Itoh. A Fuzzy-based Calligraphy System Using Fractals, *Proceeding of The 3rd European Congress on Intelligent Techniques and Soft Computing,* pages 1440-1444, 1995.

[Poggio *et* al., 1985] T. Poggio, V. Torre and C. Koch. Computational vision and regularization theory. *Nature,* Vol. 317, 26 September, pages 314-319, 1985.

[Sederberg *et al,* 1992] T.W. Sederberg, E. Greenwood. A Physically based Approach to 2-D Shape Blending. *SIGGRAPH'92, ACM Computer Graphics,* Vol. 26, No.2, pages 25-34, 1992.

[Zhang *et al,* 1993] X. Zhang, H. Ji, H. Sanada and Y. Tezuka. Forming Brush-Written HIRAGANA with the capability of Providing Versatile Stroke-Connecting Flows. *The Transactions of the Institute of Electronics, Information and Communication Engineers,* Vol. J76-D-II, No. 9, pages 1868-1877, 1993.