Example-based Design of Artistic Calligraphy Fonts

Yutaka Goda  
Okuma Corporation  
Oguchi-cho, Niwa-gun, Aichi, Japan

Tsuyoshi Nakamura  
Nagoya Institute of Technology  
Gokiso-cho, Showa-ku, Nagoya, Japan  
Email: tnaka@nitech.ac.jp

Masayoshi Kanoh  
Chukyo University  
101 Tokodachi, Kaizu-cho, Toyota, Japan  
Email: mkanoh@sist.chukyo-u.ac.jp

Abstract—“Shodo”, which is Japanese calligraphy, is a form of artistic writing used for writing the Japanese and Chinese language. This study proposes a method of designing Japanese calligraphic fonts which contain artistic representations similar to actual Shodo art. We especially focus on kasure (a Japanese term referring to the breakup of brushstrokes), which is one of the most characteristic visual effects of Shodo, adopt an approach using example-based texture-transfer to achieve high quality artistic fonts.

I. INTRODUCTION

There has recently been interest in research into non-photorealistic rendering (NPR) which aims to convert photographic images into computer graphic (CG) images in the style of paintings, such as watercolors or oils, to reproduce CG images that imitate painting styles as if drawn by an artist. Methods of providing painting-style conversion can be divided into two main categories: those that use image filters and those that retrieve texture patches from source images to form synthesis images (Texture Synthesis and Texture Transfer). The latter methods make it easy to reproduce the texture of an original painting since they use parts of existing painted images as texture patches.

Calligraphy fonts are an example of how the artistic forms of Shodo are being brought to life in everyday life. These calligraphy fonts are often used in DTP and in the printing of New Year’s cards by individual users, and it is certainly not uncommon for people to use calligraphy fonts to create prints. It is considered that the design of calligraphy fonts can be included within NPR research in its broadest meaning, and up until the present there have been a number of reports on research relating to the creation and processing of calligraphy fonts. Typical research into this field includes techniques of representing the impression of kasure in calligraphy fonts by using brush-feel cursors, mathematical morphology, but this is considered to be equivalent to the previously described approach of using image filters in research into painting-style conversions. In a similar manner to paintings, there is a large number of celebrated artworks in the Shodo genre.

We propose a method of using existing Shodo artwork as source examples and use the esthetic representations in that artwork in the redesign of calligraphy fonts. It is expected that it might be possible to implement the reproduction of the high level of texturing that is characteristic of existing Shodo artwork, in a similar manner to painting-style conversion. In other words, we can acquire output that reflects the kasure features of a source character image by considering the inputs of a target font that is the object to be processed and the source character image that is a source example, as shown in Fig.1.

The contributions of this study to calligraphy font design are as follows:

1) Focus on “kasure representation” which is one of the features of existing Shodo artwork, by the method proposed in this paper that uses source examples, to help in the creation of a calligraphy font that reproduces such features.

2) Facilitating the design of calligraphy fonts with kasure representation by using source examples, without requiring the user to have specialist knowledge, a sophisticated artistic sense, or even skill at such difficult operations.

3) Enabling of the implementation of different kasure representations even when the target character is the same, by simply modifying the source examples. This can be expected to enable the representation of various different kasure effects, in comparison with conventional methods that use brush-feel cursors or mathematical morphology.

It is of course also be possible to apply the same source example to a number of target fonts.

Note that there are already examples of research into aspects of calligraphy character writing, including research into the outlines of fonts themselves, such as stroke thickness and the balance of character images, but those factors are outside of the scope of this study. In addition, a large variety of design and processing options can be produced if generic graphic tools such as Adobe Illustrator or photo-retouching tools such as PhotoShop are used. However, the processes enabled by such tools have problems in that they require sophisticated operating skills and an artistic sense, and it is difficult to implement highly artistic representation of actual Shodo artworks.

II. PROPOSED METHOD

Paintings and Shodo works are formed of strokes drawn with a brush, with the artwork being completed by a number of strokes overlaying each other. With Shodo in particular, the flow of strokes is interrupted as they cross or overlap each other, but away from those interruptions in the flow, the directions of the brush are comparatively clear. From observations of the kasure (broken) portions, we can see how that kasure are created along the flow of the brushwork. In addition, if the Shodo artwork itself is perceived as textures,
we could also consider that the flow of the brushwork could be represented by sequences of groups of small texture patches.

For that reason, this study first gives character image skeletons of a source character image that is existing Shodo artwork and a target font that is an existing calligraphy font, as graph structures \( G_S \) and \( G_T \). This makes it possible to specify that the brush flow follows the edges of each graph that is a character image skeleton. Fig.2 shows examples in which the source character image and the target font that were shown by way of example in Fig.1 are applied to the graph structures. Each graph is given terminal nodes at the initial and final brush portions of the brush and non-terminal nodes at places where strokes cross each other, and those nodes are linked by non-directional edges. The edges of \( G_S \) are called source edges and the edges of \( G_T \) are called target edges. These source and target edges are made to correspond in a suitable manner and, if a texture patch group associated with a source edge can be placed continuously on a target edge while maintaining its sequential relationship, something similar to the kasure representation on the source edges can be reproduced on the target edge. If this is done for all of the target edges, it is possible to acquire an output that possesses the kasure features of the source character image.

Note that this method requires three items as input, as shown in Fig.1: the target font, the source character image, and also a mapped source character image. The mapped source character image is a region which exhibits places where there is ink on the source character image and places that are equivalent to kasure (places where the brush was assumed to pass but no ink was deposited), which is used to clarify the range over which the brush moved over the paper and thus acquire a character image skeleton of the source character image. The mapped source character image must be obtained manually by the user beforehand. In this study, we used the Hilditch thinning algorithm\[11\] as the method of acquiring a character image skeleton. Details of the proposed method are given below.

### A. Graph Construction

As shown in Fig.2, the source graph \( G_S \) and the target graph \( G_T \) are formed of edges that link nodes to each other. \( v_i^S \) and \( v_i^T \) are nodes that construct \( G_S \) and \( G_T \) respectively. There are two types of node: terminal nodes \((tn)\) and non-terminal nodes \((ntn)\). The set of nodes possessed by \( G_S \) is denoted by \( V_S \) and the set of nodes possessed by \( G_T \) is denoted by \( V_T \). The edges of \( G_S \) and \( G_T \) are denoted by \( e_i^S \) and \( e_j^T \) respectively, and each edge is assigned three types of attribute \((Style, Direction, \text{and Length})\). \( Style \) is the type of edge, \( Direction \) is an approximation vector of the edge, and \( Length \) is the length of the edge. We determine the degree of similarity of edges from the three attributes assigned to each edge, and link the instances of \( e_i^S \) and \( e_j^T \) that have a high degree of similarity for use in texture assignment. The sets of edges of \( G_S \) and \( G_T \) are denoted by \( E_S \) and \( E_T \) respectively. In addition, we attempted to use this to implement a correspondence between substantially similar portions of the edges of the left-ending stroke that includes a brushstroke-end part in the character 天 ("heaven") and the edges of the left-ending stroke that includes a brushstroke-end part in the character 大 ("big"), as shown in Fig.2.

- **Edge type: Style**

  There are three types of \( Style \): edges linking illustrated in Eq.(1). Note that in this study, \((tn, ntn)\) and \((ntn, tn)\) have the same \( Style \) and no distinction is made between them in the handling, to enable the handling of non-directional graphs.

  \[
  Style = \begin{cases} 
  (tn, tn) \\
  (tn, ntn) \\
  (ntn, ntn)
  \end{cases}
  \]

- **Edge approximation vector: Direction**

  \( Direction \) is a vector represents a line segment which connects both ends of an edge. It is an attribute for evaluating the direction of the edge. In this case, we define that the Direction of \( e_i^S \) is \( \vec{e}_i^S \) and the Direction of \( e_j^T \) is \( \vec{e}_j^T \).

- **Edge length: Length**

  \( Length \) denotes the number of pixels that form an edge. We define that the Length of \( e_i^S \) is \( |e_i^S| \), and the Length of edge \( e_j^T \) is \( |e_j^T| \).
B. Assignment of Edges

One suitable source edge $e_i^S$ is assigned to the target edge $e_j^T$ (Fig.3). This procedure is described below.

1) We evaluate the degree of similarity of Style of all source edges with respect to the Style of $e_j^T$.

2) We then evaluate Direction and Length with respect to the source edges of the assignment candidates, to determine a unique $e_i^S$ to be assigned.

3) Steps 1. and 2. are repeated for all the target edges, to determine the corresponding values of $e_i^S$.

![Fig. 3. Edge assignment processing](image)

- Evaluation of degree of similarity of Style

We substitute $tn = 0$ and $ntn = 1$ to denote each type of node (Eq.(2)). In this case, we evaluate the Hamming distance between the Style possessed by the source edge $e_i^S$ and the Style possessed by the target edge $e_j^T$ as the degree of similarity between them, and are left with the value of $e_i^S$ that has the shortest distance with respect to $e_j^T$ as the candidate.

$$\text{Style} = \begin{cases} (tn = 0, tn = 0) \\ (tn = 0, ntn = 1) \\ (ntn = 1, ntn = 1) \end{cases}$$

(2)

- Evaluation of Direction and Length

We determine the source edge $e_i^S$ by using an evaluation function (Eq.(3)) with respect to each candidate obtained by the evaluation of the degree of similarity of Style. In this case, $w_r$ and $w_l$ in the equations are positive real numbers that denote weights, where $w_r$ is the weight for Direction and $w_l$ is the weight for Length. Note that we define that $w_r + w_l = 1.0$.

In addition, $r_{ij}$ (Eq.(4)) and $l_{ij}$ (Eq.(5)) both denote evaluated values of $e_i^S$ with respect to the target edge $e_j^T$, where $r_{ij}$ evaluates the angle subtended by the approximation vectors of $e_i^S$ and $e_j^T$, to evaluate the similarity of directions between the edges, and $l_{ij}$ evaluates the difference in edge lengths between $e_i^S$ and $e_j^T$.

$$\arg \max_i (w_r r_{ij} + w_l l_{ij})$$

(3)

$$r_{ij} = \frac{1}{2} \left( \frac{e_i^S \cdot e_j^T}{|e_i^S||e_j^T|} + 1.0 \right)$$

(4)

$$l_{ij} = 1.0 - \frac{(|e_i^S| - |e_j^T|)^2}{\sum_j (|e_i^S| - |e_j^T|)^2}$$

(5)

C. Assignment of Textures

We implement the kasure representation by placing the textures of the source edge $e_i^S$ onto the target edge $e_j^T$ (Fig.4). First of all, we sample $e_i^S$ at uniform spacing and form a texture patch $P_k^S$ from a region surrounding the center of the $k$-th sample point. Similarly, we sample $e_j^T$ at uniform spacing to obtain sample points. We then determine the $k$-th sample point on $e_i^S$ that corresponds to the $m$-th sample point on $e_j^T$, from Eq.(6). From this correspondence relationship, we transform $P_k^S$ and place it as $P_m^T$ at the center of the $m$-th sample point on $e_j^T$.

$$k = \left\lceil m \times \frac{|e_i^S|}{|e_j^T|} \right\rceil$$

(6)

The transform during this process is expressed by Eq.(7). $S$ denotes the rotation matrix and $\alpha$ denotes the expansion ratio. In addition, $\alpha$ makes use of the thickness $d_k^S$ of the mapped source character image at the $k$-th sample point on $e_i^S$ and the thickness $d_m^T$ of the target font at the $m$-th sample point on $e_j^T$, to expand or compress $P_k^S$ to match the thickness of the target font (Eq.(8)). Note that since some overlap will occur between adjacent textures during the assignment of textures, we use texture synthesis[11] to place the textures seamlessly.

$$P^T_m = \alpha S P^S_k$$

(7)

$$\alpha = \frac{d_m^T}{d_k^S}$$

(8)

III. Experiments

We performed experiments on calligraphy font design, using the method proposed by this study. For source images, we first used two characters: 萩 (the syllable “a”) from a calligraphy font that included kasure representation and “person”) that was obtained by scanning a handwritten calligraphic character. In the target font, we used three characters: 春 (“spring”), 夏 (“summer”), and 秋 (“fall”). Output examples obtained by combining these characters are shown in Fig.5. Note that the

---

Fig. 4. Texture assignment

different oval annotations denote the representative assignment relationships between the source character images and the target font, with portions enclosed by the same type of oval showing the assignment relationships. As shown by way of example in Fig. 5, it is clear that we were able to implement various different kasure representations by combining the source character images and the target font. However, we have to admit that there are unnatural areas in some parts of the kasure representations.

We then used output examples to show the effects obtained when the weights \( w_l, w_r \) were varied. Note that we used the source character image \( \_ \) at the top right of Fig. 5 for the source character image. The left-hand column of Fig. 6 show examples that focused on Length with \( w_l = 0.9, w_r = 0.1 \), and the center column shows examples that focus on Direction with \( w_l = 0.1, w_r = 0.9 \). It is clear that differences in weight have a large effect on the kasure representations of \( \_ \) and \( \_ \). However, there were no great differences seen for \( \_ \) ("winter"), apart from the drawing of the end point. From these output examples, we can see it is possible to vary the kasure representation to a certain degree by varying the proportions of \( (w_l, w_r) \), but this effect depends on the target and the source.

Finally, output examples produced by related research are shown in the right-hand column of Fig. 6. We used texture-by-numbers of a method called Image Analogies[4][5]. This texture-by-numbers is not part of research aimed at calligraphy font design, but it uses three inputs similar to those of this study ("source image", "unfiltered source image", and "target map"), to create any desired image having features of the source image by compositing textures. For this trial, the source image, unfiltered source image, and unfiltered target image corresponded to the source character image, mapped source character image, and target font of this study, to produce output examples. It is clear from a comparison with the output examples of the proposed method that since the continuity of brush flow is lost, an unnatural kasure representation is produced thereby. Thus texture-by-numbers cannot provide processing that gives graph structures like those of this study, which is thought to be due to the fact that the sequential relationships of texture patches in the source character image could not be preserved.

IV. CONCLUSIONS

In this paper, we proposed a method of acquiring kasure representation features from existing Shodo artwork as textures then using those textures in the processing of calligraphy fonts, and gave output examples. This method enables any kasure representation that the user desires, by just modifying the Shodo artwork. However, it is necessary to reconsider details such as the method of evaluating the degree of similarity of edges because implausible kasure representations appeared in some of the output examples. In addition to that, quality of output fonts depends on source examples that are Shodo works. Practical calligraphy font design would require many kinds of source examples that are famous old artworks, modern artists’ works and so on. In the future, we consider we should construct an integrated environment for calligraphy font design, while improving on these challenges, and we would also like to review the application of this method to objects other than calligraphy fonts.

ACKNOWLEDGMENTS

This study is partly supported by TAF(Telecommunications Advancement Foundation) and Hayao Nakayama Foundation for Science & Technology and culture.

REFERENCES

Fig. 5. Output examples ($w_l = 0.5$, $w_r = 0.5$)
Fig. 6. Comparative examples (Left: $w_y = 0.9, w_x = 0.1$, Center: $w_y = 0.1, w_x = 0.9$, Right: Texture by Numbers)