グラフィクスとＣＡＤ　76－8
（１９９５．８．１８）

毛筆フォントとCG描画技術の統合

郭　清蓮（かく　せいれん）

ソニーシステムデザイン株

コンピュータ毛筆システムの現状について分析し，この分野における問題点を指摘する．さらに，我々が開発したリアルな毛筆文字生成システムを紹介する．

An Integration of Font and CG Techniques

Qinglian Guo

Sony Systems Design Corporation

This article presents an overview on the situation of computer calligraphy software, to indicate what we think is the problem with this subject, and introduces our approach on developing an interactive system for generating realistic calligraphy words.

1. Introduction

Recently, there has been remarkable increasing interest in computer calligraphy systems Because of the popularity of word processors, the techniques of desk top publishing, the development of hardware, and the researches on outline fonts. Especially, Asia people are interested in these systems, since calligraphy is an important part of their art and culture. Nowadays, there are a few young people who can write good calligraphy words. It is urgently demanded for redeveloping and preserving traditional techniques of calligraphy in digitize formation. In Japan, some calligraphy systems are now used for creating new years cards, they have provided many benefits to people who want to create good formation calligraphy words. However, professional people, who have knowledge of calligraphy, are not satisfied with the outputs of these systems. Because the systems only fill in the inside of outlined word uniformly with monocular and it is impossible to represent the rendering effects of painting materials.

Fig 1. gives a comparison. A calligraphy word on the left is created by using a computer, and two calligraphy words on the right are artists' works. It is clear that the artists' works are much more impressive because of the texture of ink breaking up on the strokes. We call this type texture as "Kasure". Kasure is usually used to show the breath, the strength, and the speed of moving a brush. Since ink is merely gray tone, artists often use these techniques to furnish the strokes with life and strength, to give a dynamic and impressive appearance to communicate his understanding to the meaning of the word. The fact that the existing systems can not represent this rendering effect is a big drawback which makes the output unnatural and limited to lowest level.

On the other hand, in the area of CG, reports exist on simulating the rendering effects of painting materials. The most notable research is Strassmann's "Hairy Brushes" [3]. His work provided a valuable base for afterward researches on modeling brush works. Since brush is analyzed physically, his model results in images similar to drawing of real brushes. Guo [1] presented a basic idea to synthesize the texture of ink diffusion onto absorbent paper. The algorithms for representing the fabric structure of paper and for modeling the dynamic process of ink diffusion is effective enough for producing natural and impressive brush drawings.

Under this situation, we started an approach to apply these CG techniques to the generation of calligraphy words. Our purpose is to include reality and flexibility to computer generated calligraphy words. In this article, we will go through the system and enter details of some parts relating with font, parameters, and rendering process.

2. The construction of our calligraphy system

In fig.2, the construction of our system is illustrated. The system has three characteristics: (1)

we integrated computer graphics techniques to font for digitize representing the effects of brushes and paper. As shown in fig. 3, our result is very impressive and realistic; (2) we proposed a new concept, i.e. adding rendering parameters to font. By this way, it is possible to involve flexibility and variety to calligraphy words. As illustrated in fig. 4, from single font data, we can make many different results; (3) we have a visualized human interface, which supports for creative and exciting ways of generating calligraphy words. It is enable users to adjust formation of words and to specify rendering parameters easily. Fig.5 gives an example of operation through the interface.

3. Font data generation processor

For representing "how a brush is moved when drawing a word", we proposed a new outline font format called "brush-stroke font". It differs from existing outline font in: (a) one of the outline points is specified as starting point / ending point, which is used to illustrate the direction of moving brush; (b) each word consists of several strokes, and there is a sequence among these strokes; (c) each stroke is defined to just correspond to a "stroke" of practical calligraphy. This differs from the existing outline font in which one stroke may cover several "strokes".

We developed an interactive font designing tool for generating font data. It is possible to create user's own handwriting font, and also to transform existing outline font to the brush-stroke font. The transformation is difficult to be performed automatically, because brush-stroke font contains more information. Designer's service is necessary for determining an intersection of strokes. In our tool, there are some special functions, such as setting starting point,picking up a controlling point of outline font, adding new point, and changing vector data. When a word of outline font is visually illustrated on a display, it can be used as a reference for designing the corresponding brush-stroke font. It is easy to divide a complex unit into strokes by using these special functions. As a result of transformation,stroke-font data will cost more memory space since it contains new control points and some duplicated control points.

4. Rendering parameters

With respect to rendering parameters, we have two groups, i.e. brush data and kasure parameters. In the brush data group, a brush is represented as an array of bristles. Each bristle has its specified ink density and ink quantity. When rendering a stroke, each bristle will correspond to a trajectory on the stroke. The kasure parameter group decides the position, density and randomness of kasure texture. As show in fig.6,

P1 and P2 control the length and position of kasure texture

P3 and P4 control the width and position of kasure texture

P5 and P6 control the random variation of kasure texture at start position/end position

p7 controls global density of kasure texture

The interface is significant in that the rendering parameters could be controlled in several ways. the simplest way for new users is to use an auto-rendering model, or to select a rendering texture from the imaged icons. For the people who are used to the system, mouse-adjustment-menu can be used to intentionally control the rendering parameters.

Kasure is realized related with stroke formation in great deal. From stroke data, it is easy to calculate stroke formation features, such as stroke length, variation of stroke width, variation of stroke curvature, and the number of positions where change in curvature is lager enough to cause a change in stroke laying direction (we call this positions as a turning position). According to these features, we classified the strokes of calligraphy words into 20 types based on statistical investigation and study about several calligraphy education books [2]. For each type, the most usually used kasure texture is designed and the corresponding parameter values are prepared in the system.

5. The process of rendering strokes

When rendering a word, we are required to decide which stroke should be rendered with kasure texture. A good balance could be obtained when long strokes are selected. A word is rendered when all its strokes are rendered. For a single stroke. we first interpolate the outline data for obtaining a smoothed boundary, then divide the area within the boundary into polygons. We could get a rendered stroke when all the polygons are rendered.

Let's see how a polygon is rendered. First, determine the intensities of the pixels on its front and back edge according to brush data. Second, determine the intensities of the pixels inside the polygon from the front and back edge. When kasure parameters are specified, we have to perform more calculation. It is necessary to decide start position Ks and end position Ke of kasure area for each rendering trajectory. For doing this, we used following relationships:

$$n = P7\,|P3 - P4|\,m$$

$$Ks\,(j) = P1 + \mathrm{random}(P5)$$

$$Ke\,(j) = P2 + \mathrm{random}(P6)$$

here, random functions are important for making a natural texture. Suppose that Li is the distance of ith polygon from starting point. When $Ks(j) < Li < Ke(j)$, we will change the color of the points on the front edge and back edge corresponding to jth trajectory to white color. Then, the points on this trajectory within the range of this polygon will all be white color. By this way, kasure texture is produced in the process of rendering the stroke.

6. Conclusions

The system is implemented in C and runs at interactive speed on a Sony NEWS workstation, without additional hardware requirement. The system is quite successful in assisting users to easily produce a variety of calligraphy words. Although a word drawn by an artist may be finer than ours. However, considering that our painting was made on a computer by a person who is not skilled at calligraphy in a few minutes , we find out results very encouraging.

References

[1] Qinglian Guo and T.L. Kunii, ``Modeling diffuse paintings of `Nijimi'," Proc.IFIP WG 5.10 Working Conference (Tokyo, Japan), 329-338, May 1991.

[2] Qinglian Guo, "Generating realistic calligraphy words", Proceeding of the Sixth International Conference on Human-Computer Interaction,(HCI International '95), Tokyo, Japan, 9-14, July, 1995, Volume 2, 129-134.

[3] S. Strassmann, ``Hairy brushes," Computer Graphics, vol.20, no.4,pp.225-232, Aug. 1986.

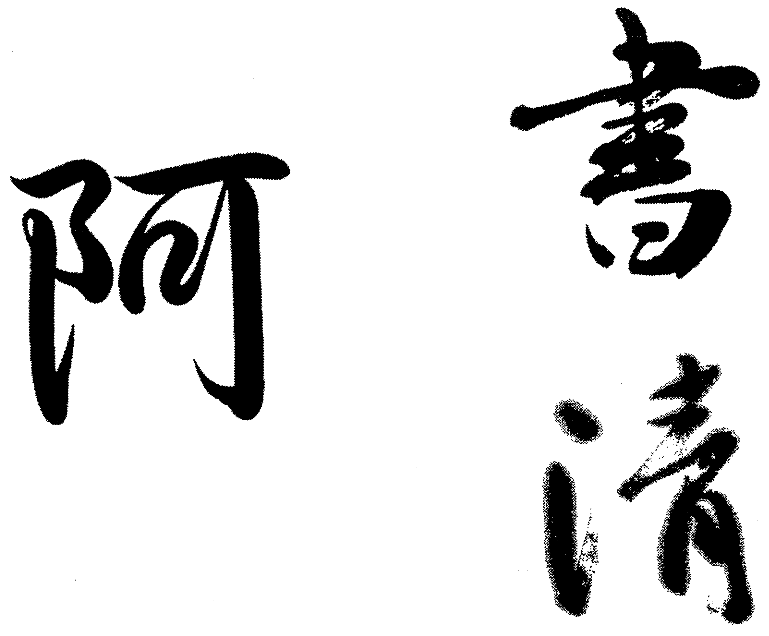

Fig.1 A comparison between output of existing computer system and artists' works

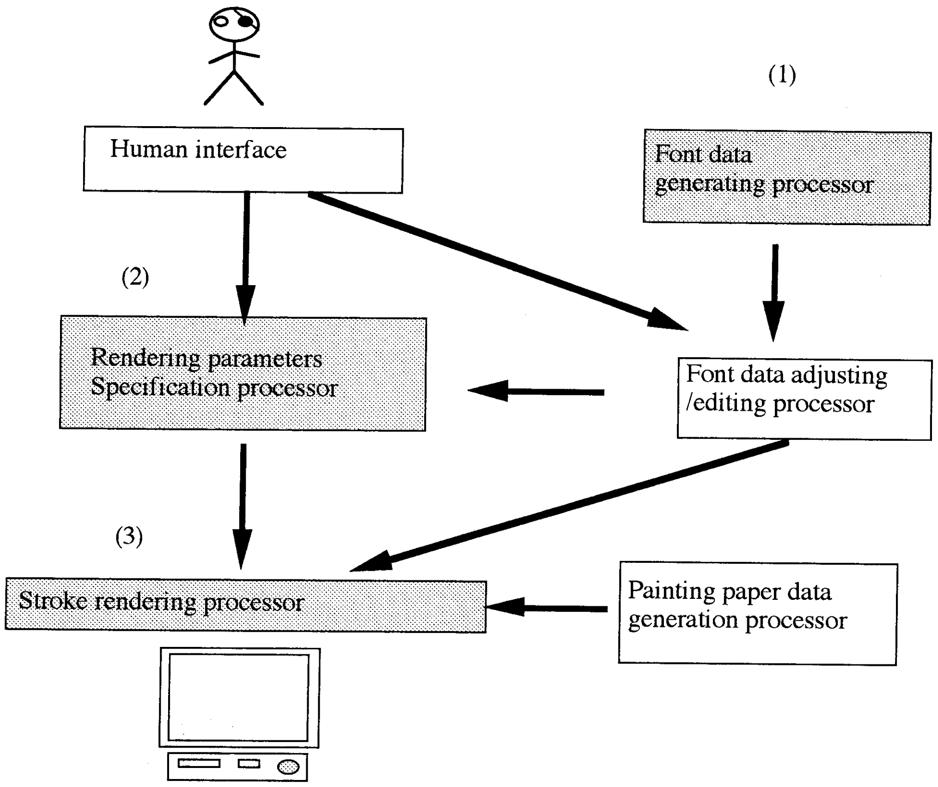

Fig.2 The construction of our calligraphy words generation system

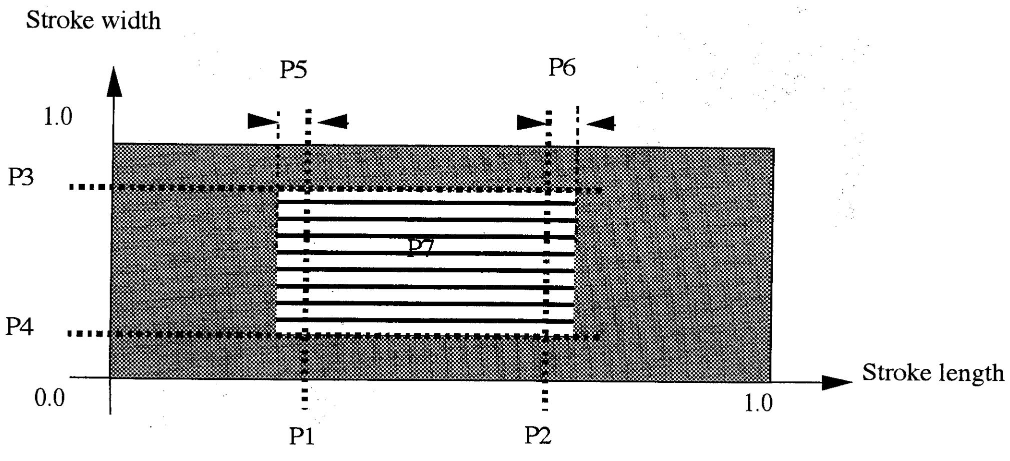

Fig.6 Kasure parameters

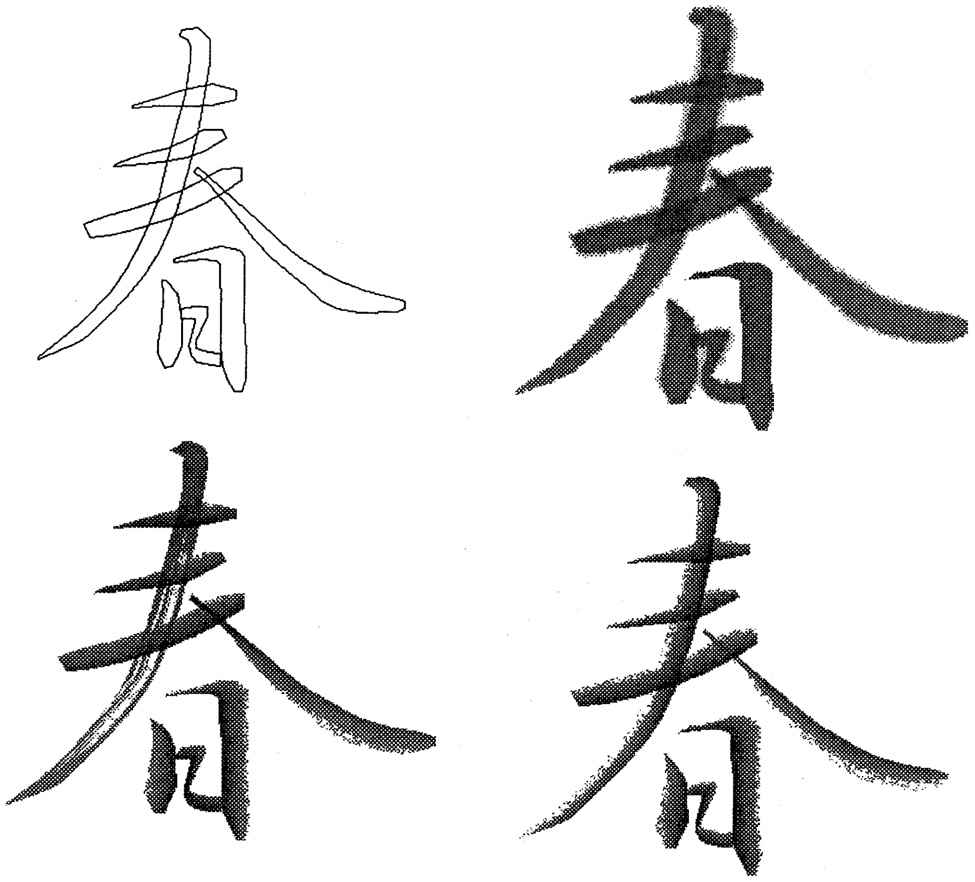
Fig.4 An example of different outputs based on the same font data

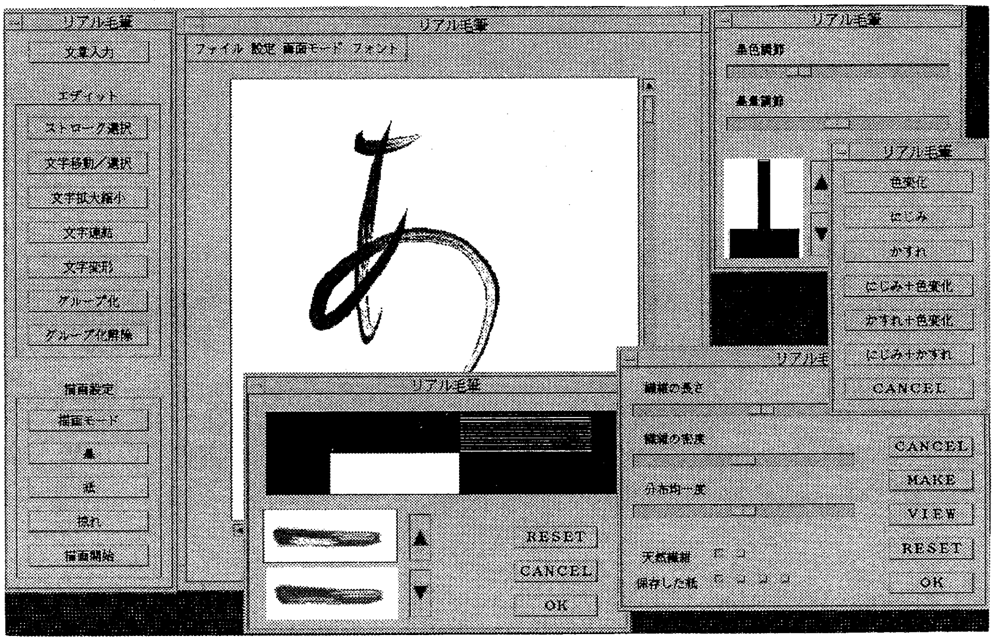

Fig.5 The human interface of our system

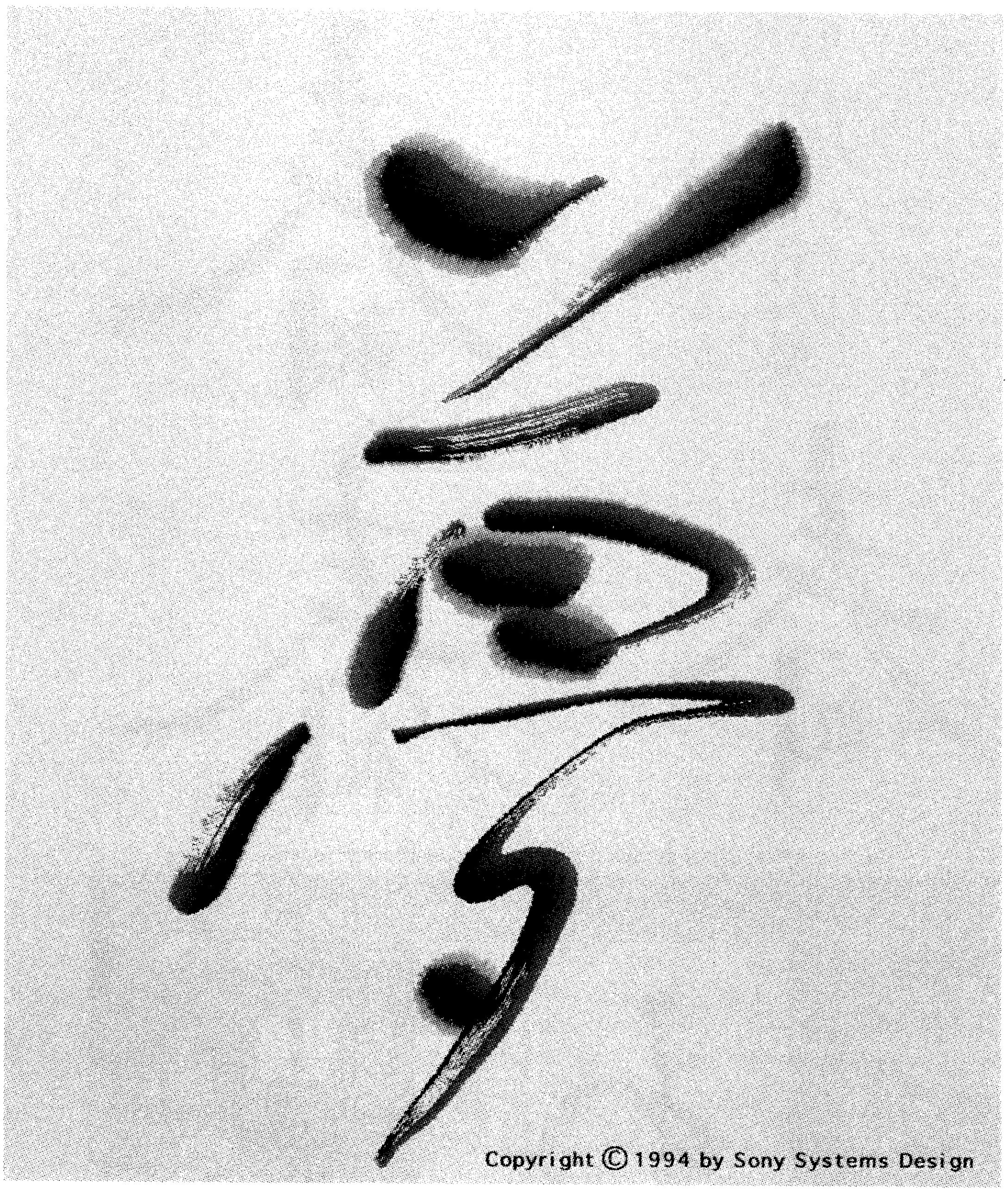

Fig.3 An output of our calligraphy system