國立交通大學

資訊科學系

碩 士 論 文

特定中國書法風格之知識推論模擬

Knowledge-Based Synthesis for Specific Chinese Calligraphic Style

研 究 生：江玠峰

指導教授：施仁忠 教授

中 華 民 國 九 十 三 年 六 月

特定中國書法風格之知識推論模擬

Knowledge Based Synthesis
for Specific Chinese Calligraphic Style

研 究 生：江玠峰　　Student：Jie-Feng Jiang

指導教授：施仁忠　　Advisor：Zen-Chung Shih

國 立 交 通 大 學
資 訊 科 學 系
碩 士 論 文

A Thesis
Submitted to Department of Computer and Information Science
College of Electrical Engineering and Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Computer and Information Science

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

特定中國書法風格之知識推論模擬

研究生: 江玠峰　　　　　　　　　　　指導教授: 施仁忠 教授

國立交通大學資訊科學系

摘　要

在中國文化藝術中，書法是極為重要的一環，其中又以各個朝代的多位書法名家最廣為大家熟悉，如：顏真卿、柳公權等。書法家利用獨特的書寫工具－毛筆，依據中國漢字造型的特性，透過藝術構思，寫出不同的筆觸，形成各自獨特的書法風格。本篇論文的目的，便是利用電腦合成技術，模擬出特定書法名家筆下的真跡。我們結合了毛筆模組以及專家系統，將書法家筆下的各種筆法加以分類，設計完善的推論規則，產生出和書法家極為相似的書法字。同時配合手寫板的輸入方式，建立一個可供使用者便利地臨摹中國書法字的系統。

Knowledge-Based Synthesis for Specific Chinese Calligraphic Style

Student: Jie-Feng Jiang Advisor: Dr. Zen-Chung Shih

Department of Computer and Information Science

National Chiao-Tung University

ABSTRACT

The Chinese calligraphy plays an important role in Chinese art. There are many famous calligraphers in different dynasties, such as Yan Zhen-Qing (顏真卿) or Liu Gong-Quan (柳公權). They use the Chinese brush stroke to generate Chinese characters with different writing styles expressing their unique artistry. The purpose of this thesis is to synthesize the specific Chinese calligraphic style of one of the famous calligraphers mentioned before. We integrate the Chinese brush stroke model with expert system, classifying all the stroke types, designing complete inference rules, to generate Chinese characters that are closely similar to those written by the calligrapher. We also use tablet pen as an input device to create a convenient interface for users to write by imitating copybooks for calligraphy.

Acknowledgements

First of all, I would like to express much respect to my advisor, Dr. Zen-Chung Shih. I greatly appreciate his patient teaching and guidance. His fine instructions made me accomplish this thesis at last. Moreover, I want to dedicate the achievement of this work to my family and my girl friend. They encouraged me and gave me support during these days. I can feel love from them. Besides, I am grateful to all members in Computer Graphics and Virtual Reality Laboratory for their help in these days, I deeply treasure the friendship among us. Last, many thanks are gave to all other people who have ever helped me on my research.

Contents

List of Figures

Chapter 1

Introduction

The Chinese calligraphy is one of the quintessence in Chinese art. Calligraphers use the Chinese brush stroke to write glamorous characters. As the brush moves, we can see the affections of the calligrapher/artist and his unique artistry. In other words, calligraphers express their attitude of mind and values by writing characters.

There are several categories in Chinese calligraphy, for examples, Kai (regular script), Li (clerical script), Xing (running style), etc. Among all these topics of Chinese calligraphy, Kai is the most common one. Many calligraphers are famous in writing Kai. Yan Zhen-Qing(顏真卿)'s Kai looked handsome and dignified, and represent imposing manner. Liu Gong-Quan(柳公權)'s Kai is thin and tall, implying strength, energy, and power. Figure 1.1 shows these masters' authentic manuscripts.

(a) Liu Gong-Quan's Kai style (b) Yan Zhen-Qing's Kai style

Figure 1.1 The different Kai styles written by Liu Gong-Quan [2] and Yan Zhen-Qing [3].

In Chinese calligraphy, students who want to write beautiful characters like that written by famous calligraphers should imitate the copybooks first. After practicing repeatedly, they could find out the extensive knowledge and profound scholarship of the Chinese calligraphy, and create their own styles. The purpose of this thesis is to synthesize the specific Chinese calligraphic style of one of the famous calligraphers mentioned before. Our system also provides a convenient environment for users to imitate copybooks of calligraphy by computer techniques.

1.1 Overview

There are lots of researches that aimed to generate realistic calligraphic character images. Most of them focused on the physical properties of the Chinese brush stroke. Users who want to write a calligraphic character using those systems need to have enough basic knowledge of calligraphy. And some special equipment (including the real Chinese brush stroke connected to the computers) is also need. Thus, it is not convenient for most users.

In this thesis, we propose a system to synthesize Yan's Kai fonts. Our purpose is to generate calligraphic characters that look closely similar to those written by Yan. The system progress in detail is as follows:

First, users using our system are not asked to have any idea of the Chinese calligraphy. What they have to know is the order of the strokes in one Chinese character. When imitating Yan's characters, users use the tablet pen as the input device. They can put the copybook of calligraphy as the background in the system, and follow the path of each stroke according to the stroke number order by the digital pen.

When a character consists of several strokes has been described as a set of digital pen trajectories, the system will decide which kind of stroke it suits for each stroke. Our program agent sends all attributes of each stroke to an inference engine, namely expert system. After some appropriate inferencing with complete rules we designed and set before, the expert system returns the best stroke type that we can apply on the stroke for each one. Finally, a character with calligraphic style is already generated, namely the Yan's Kai style.

1.2 Thesis Organization

The rest of this thesis is organized as follows. In chapter 2, we review the previous related researches about brush stroke model and expert system. In chapter 3, we describe the system structure by showing the system framework. We also have some talks about DRAMA – an expert system we use to do some inference. Chapter 4 states the details of each step of the system progress mentioned on the last chapter, including the stroke model we use, stroke type classification, stroke attribute analysis, and the rule design in expert system, etc. Then we show several implementation results in chapter 5 and have discussions on them. Finally, we give some conclusions on this work and address some future works to improve our work in chapter 6.

Chapter 2

Related Works

In this chapter, we are going to discuss some researches related to Chinese calligraphy. Section 2.1 states the basic unit in synthesizing calligraphy, the brush model. Some researches aim to the physical properties of it to make various kinds of simulation. Strokes are generated and form a Chinese character according to the brush model. On the other hand, most researches about Chinese calligraphy focus on the aspect of image processing, e.g. the process of keeping valuable calligraphy authentic work. And some focus on the combination of brush and image process. These will be discussed in Section 2.2. Then, we survey the researches about expert systems in Section 2.3.

2.1 Brush Model

There are a lot of researches that aim to the topic of brush model. Brushing commonly refers to the drawing of curves with various line widths in bit-mapped graphical systems. In 1989, Porch and Fellner proposed the Circle-Brush Algorithm [13]. This approach produces constant line width by circles of suitable diameter, independent of the curve's slope.

At the same time, Strassmann's hairy brushes model [16] was presented. It provided a description of physical properties of brush materials in order to generate hairy brush images of sumi-e paintings – one kind of traditional Japanese art. And the concept of a collection of bristles was proposed for the first time.

After that, Horace and Helena [10] proposed the first methodology for generating hairy-brush writings. A parameterized model is built to specify the varying brush orientation and brush tip pressure, the brush writing hair properties and the variation of ink deposition along a stroke trajectory. From this model, people can simulate the physical process of brush stroke creation and synthesize most of the aesthetic features of calligraphic writings. This method is suitable for Chinese calligraphy.

Then, Nelson et al. [12] presented a 3D brush model. The main feature of this model is its ability to mimic brush flattening and bristle spreading due to brush bending and lateral friction exerted by the paper surface during the painting process. Since the visual feedback is significant in their system, some special equipment is required such as a real brush and haptic device.

2.2 Image Processing on Calligraphy

Most researches about Chinese calligraphy are devoted to image process of the calligraphic documents. Yang et al. [18] proposed a method to vectorize the digital images of the Chinese characters automatically. The vectorization results can be transformed into a true-type font for general applications. They also prevent the zigzag phenomena when enlarging the characters. In this way, the treasures of the arts of Chinese culture can be preserved.

A different kind of application on calligraphy and image processing was suggested by Wei et al [17]. They proposed a method to generate scratched look calligraphy characters by mathematical morphology. By this method, people can decide on number of times of the thinning computation and structuring elements, and can also know whether the sizes of generated calligraphy characters are the same as the original one in theory.

A combination of brush and image process was presented by Mi et al [11]. They proposed a virtual brush model based on droplet operation and its application on retrieving character outlines and character modeling in Chinese calligraphy style. The droplet model helps to compute stroke area with well-defined geometry information and leads to the feasibility of retrieving the outlines of characters with well-defined geometry representation.

2.3 Expert Systems

By definition, expert system is a system which employs human expertise to solve problems like the real mankind does. An expert system either supports or automates decision making in an area of which experts perform better than non-experts.

Tzeng et al. [4] introduced the concept of "knowledge" used in expert system. There are four main topics: knowledge representation, knowledge acquisition, knowledge inference, and knowledge validation/verification. By these topics, we can choose the suitable method for each topic to form our calligraphy system.

In the first topic, many representative mechanisms are used for knowledge representation. Backus–Naur form is the most common one used as a meta-language for defining the syntax of a language. Frame-based representation is suitable for related knowledge about a narrow subject with much default knowledge. Case-based representation is usually used to describe the experimental knowledge, and to solve problems by the experience of old cases which are similar to the current one. Here in our calligraphy system, we use rule-based knowledge for the knowledge representation. This representation is appropriate for a kind of knowledge field which requires inference. The basic rule form is if-else condition. With complete rules, all situations can be inferred.

The goal of knowledge acquisition is to elicit expertise from domain experts. We reference the calligraphy copybook of Yan's Kai style for increasing the domain knowledge of Chinese calligraphy. Then, a repertory grid is needed for knowledge acquisition. In this method, we elicit elements to be classified. Within the modify stage, each grid is rated by filling a rating. At last, the repertory grid contains the state of the attributes in it. By this way, all rules can be generated from the grid. We show the final rule set in appendix to express the concept of repertory grid.

The third topic is the method of inference which has a main theory called syllogism. It is a simple, well-understood branch of logic that can be completely proven. In general, a syllogism can be any valid deductive argument having two premises and a conclusion. Using the syllogism with two ways of reasoning – forward and backward, the decision of any situation can be made from the input conditions or from the final conclusions on the contrary, respectively.

In the last topic, knowledge should be verified and validated. Most expert system has non-exclusive rules and fuzzy decisions. It leads to some unexpected situations. To avoid such kind of problems, we set the rules used in Chinese calligraphy mutually exclusive. Therefore, the same attribute state will not go to different conclusions.

Chapter 3

System Structure

In this chapter, we will introduce the structure of the whole calligraphy system. We propose an approach which integrates expert system into calligraphy by using different developmenting environments, including the inference mechanism (using the JAVA environment) and the result display (using the C ++ environment) of the calligraphy. The proposed system consists of 5 modules according to the developmenting environment. Figure 3.1 shows the system architecture. We are going to describe the details of each module in Section 3.1. Besides, since we combined another independent system (the 3rd module in Figure 3.1), some other descriptions will be given in Section 3.2.

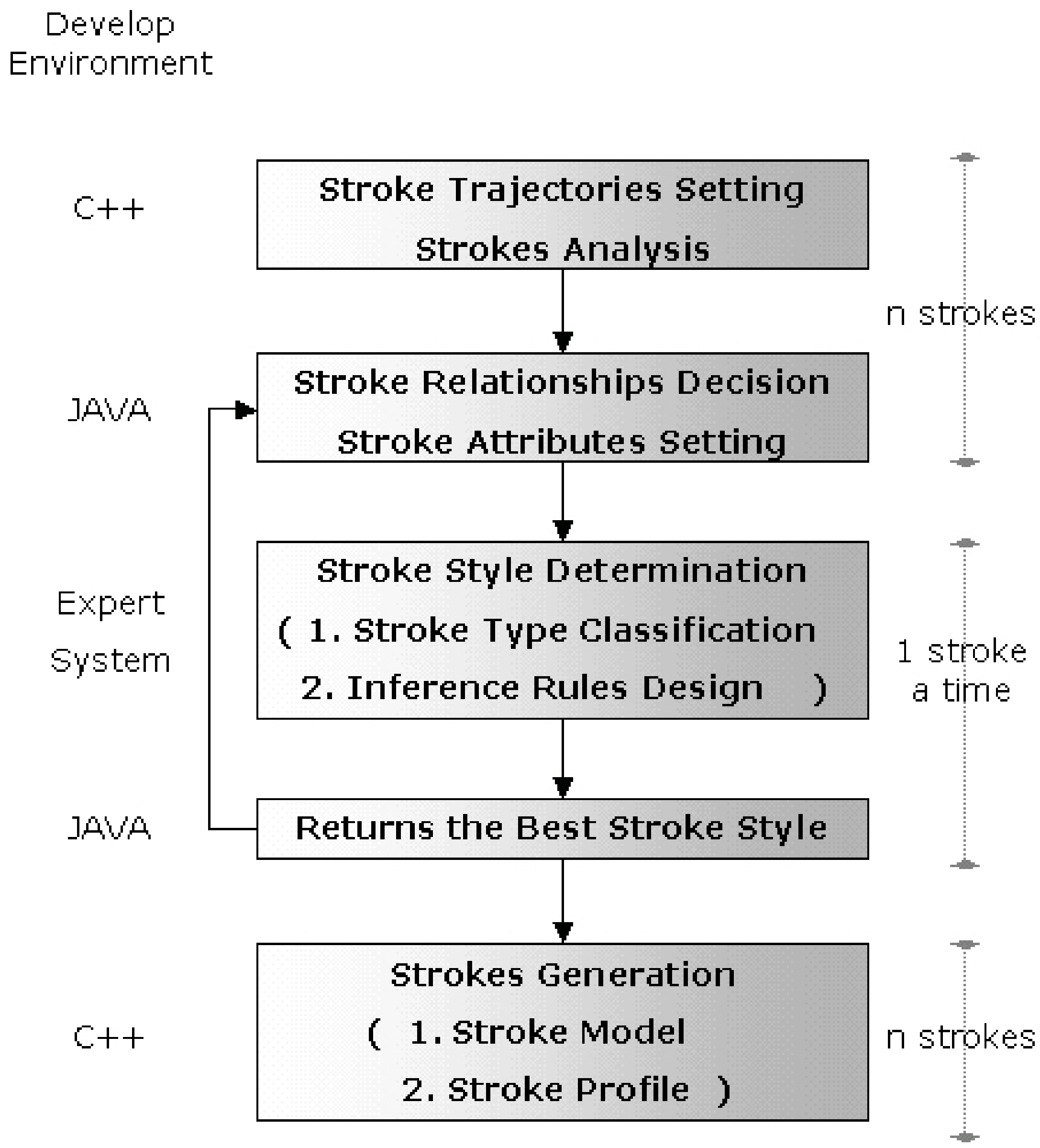

Figure 3.1 The system architecture

3.1 System Overview

The proposed calligraphy system consists of 5 modules: stroke trajectories setting and analysis, stroke relationships decision and attributes setting, stroke style determination, returns of the best stroke style, and the strokes generation. Among them, the first and the last one use C++ as the developmental environment, and devoted to input and output. The others (the middle three modules) are responsible for inference mechanism and are developed in JAVA environment.

In the first module, we input the stroke trajectories which can be done with a digitizing pen or a mouse to start the calligraphy system. In this part, our approach is to imitate the calligraphic copybook of Yan's. With the graphic API – OpenGL, we set the size of the copybook be 400 x 400 pixels and 256 gray level for each pixel. It is put on the screen as the background and the user can write Chinese characters in it. When a character has been written, the first step of this module is accomplished at the beginning. We can only see the mouse trajectories of the character by removing the copybook. After that, the second step of this module can be carried on: strokes analysis. The system will analyze every stroke roughly in this stage, and determine the type of each stroke.

In the second module, the relationships among all strokes are determined. According to the mouse trajectories that are set in the previous module, we can measure the starting and ending positions of each stroke, check whether the strokes of a character are connected or joined to each other, and calculate more information of each stroke. Recording all these data set, we can enter the inference mechanism then.

In the previous two modules described before, all information of each stroke has been already collected. Now, we are going to determine the stroke style. There are two sub-routines here. First, we have categorized all strokes and arranged them in much more kinds in terms of the writing style of Yan and the characteristics of his calligraphic character. This procedure is named stroke type classification. Then, we design the inference rules according to the stroke information collected before and use expert system as the inference engine. Via these two sub-steps, the style of each stroke can be determined.

After the inference, the fourth module handles the return of it. Because the system has crossed different developmenting environment – JAVA and C++, we utilize transmission of file in this part, and make them (JAVA and C++) communicate to each other correctly.

Finally, when each stroke is allocated with a specific stroke type, we can construct the whole character and show the result in the fifth module. The graphic

API – OpenGL is used in this module. We use glDrawPixels to render the calligraphic character with specific stroke model and stroke profiles. And this character will appear in Yan's style at last.

3.2 DRAMA

In the last chapter, we have discussed expert systems about the four main topics on "knowledge" in detail. Here in this section, the focus is put on the developmenting environment – DRAMA, a JAVA-based expert system.

DRAMA is a rule-based expert platform. As to the general engineers who construct the knowledge base, this is an intact knowledge structure that is easier to use. By adding rules that are similar to human logical thinking [8] [14], DRAMA can have the ability of knowledge inference, and offer service for lots of different knowledge field.

There is an important design concept in DRAMA – NORM. It stands for "new object-oriented rule model" [7] [15]. By the concept of object-oriented knowledge management, this kind of system can help knowledge constructors, the engineers, to develop knowledge applications in a manner closer to human thinking. The purpose of this model is to consider the psychological aspect of real mankind.

This consideration is often neglected by traditional expert system which just simply use procedural knowledge as the system basis. This makes the description of knowledge often lack the accuracy. However, the system with NORM model takes some characteristics of human logical thinking into account, to design different knowledge fields related to each other.

In NORM model, knowledge class is designed to perform knowledge block or knowledge concept. We have some operations moving among different knowledge classes:

1. When a new knowledge is constructed after learning some expertise, we usually connect the current knowledge with others which we have learned before, to reduce the difficulties of learning the current one. This is called “reference”.

2. Knowledge “modification” is a kind of operation we are familiar with. The learner can increase learning efficiency by some well-modified knowledge. Besides, “extension” is also important on knowledge construction. According to the demand for advanced knowledge, learners use existing knowledge to increase extra knowledge content.

3. When the problem in a knowledge class can be simplified or turned into another kind of knowledge problem, we use “trigger”.

4. According to different decision conditions, we need utilizing different knowledge classes to infer. At this time, “acquire” is needed. In other words, we should find out the proper knowledge classes to handle the sub-problem.

By the operations described above, the NORM model is complete and makes DRAMA more friendly to use.

Finally, we show the system structure of DRAMA in Figure 3.2. It includes calligraphy program handling I/O and processing, server and inference engine of DRAMA, and the knowledge base constructed with NORM model.

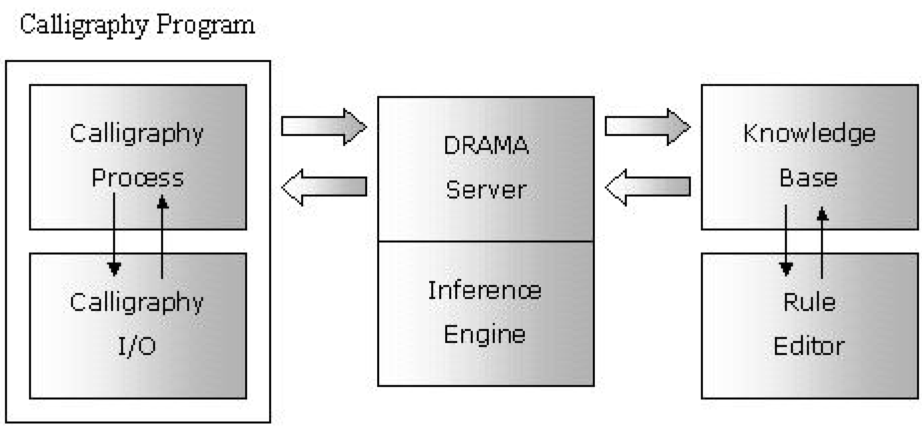

Figure 3.2 Calligraphy system

Chapter 4

Knowledge-Based Calligraphy System

Based on the system structure described in Chapter 3, in this chapter we are going to do more detailed discussion to every module. All discussions are given from the point of view of implementing the calligraphy system. The order of explanation may (1) be with the order of system procedure or (2) be with the order of system design (or implementation order). Although (1) is a more logical formulation, (2) is preferred since it is more intuitive for users to know the design procedure. Besides, we can know how modules are linked to each other more completely by this way.

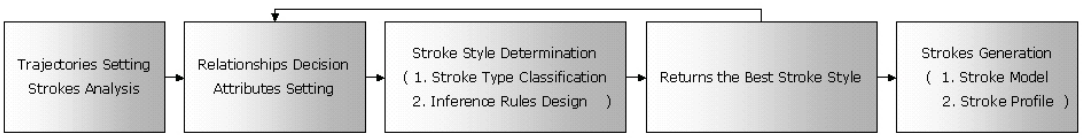

Figure 4.1 System flow chart.

4.1 Stroke Model

Stroke model is the basic component in Chinese calligraphy synthesis. In calligraphy system, it lies in the fifth module. Cooperating with setting of stroke profiles, the final character will be presented by stroke generation.

In [9], the proposed virtual brush has several features. Here, we integrate some of those features into our system as follows:

1. The modeling way allows the synthesis of gray-level calligraphic images with fine ink trails.
2. The brush pressure is ignored since the main purpose of this system is to generate characters in Yan's style by the composition of all strokes, not focusing on the interactions of each stroke that users write.
3. Realistic calligraphic effects can be produced by varying different parameters which control the profile of brush orientation along a stroke trajectory.
4. The contact between the bristles and the paper is modeled using an elliptic footprint whose principal axes can be dynamically adjusted according to different turning control.

Then, the brush model is applied. In normal state, it is formed like a brush bundle which can be approximated by an inverted cone while suitably inked, as shown in Figure 4.2. It has the necessary information relating to the geometry of the brush bundle. For example, the radius of the brush stem (R), the length of the brush bundle (L), and the number of bristles (M) that form the brush bundle.

A typical example of a real brush has the following values: the brush stem radius (R) = 0.9 cm., the length of brush bundle (L) = 3 cm., and the number of bristles (M) = 10 000 approximately.

In a digital environment, these values above should be adjusted. The radius (R) and the length (L) change based on the system size. In general case, each piece of the brush bristle paints just one pixel or more. Here in our system, the bristles are ignored because we do not focus on the simulation of stroke movement interactively. Thereby, the radius and the length dominate the stroke generation.

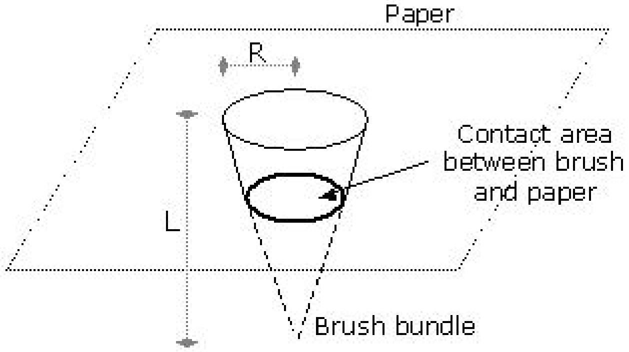

Figure 4.2 The brush bundle in an inverted cone shape.

Although the parameters (R and L) are fixed for a given brush, the radius of the contact area between the virtual brush and the paper varies as the brush model rises and falls on the paper. In other words, the larger the contact area between paper and brush stem, the more pixels it needs to fill the canvas. This effect is shown in Figure 4.3. Except for the contact radius, another factor also determines the brush effect, which is the orientation of the brush. It depends on the direction that the brush moves along and appears in an ellipsoid shape. For example, when a vertical or horizontal stroke is written, the ellipse should be a or a shape, respectively, and, for a slanting stroke, it will be a or a shape, respectively. Combining the radius of contact area and the ellipse orientation, we can achieve some special properties of the Chinese calligraphy, especially on the start or the end of each stroke. In this way, all kinds of stroke type can be generated. Figure 4.3 shows some examples.

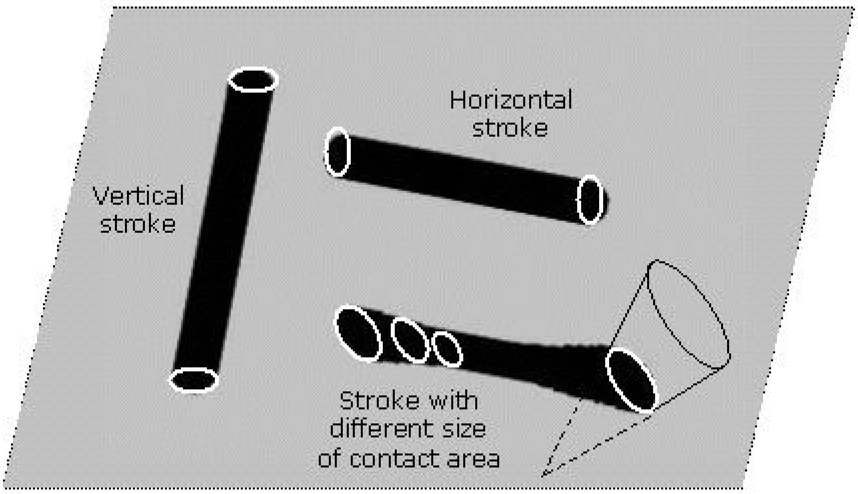

Figure 4.3 The changes of the ellipsoid contact area in the size and the orientation.

In the stroke profile, we apply the radius and orientation parameters to specify the stroke shape. The format consists of several data, including the percentage range on a stroke, the deviation degree in this range, the ellipse radius and orientation specifying the brush drops on this range. If the better result of stroke is preferred, the percentage range should be narrowed. Therefore, we can describe the stroke detail more explicitly in each range.

4.2 Stroke Type Classification

Before the progress entering the expert systems, first we should know which kind of form the result would be presented after inference. In other words, the best stroke type is chosen for the appearance of the stroke each time. This work is done by the stroke type classification, the third module in the system structure as shown in Figure 4.1.

In the classification of the Chinese characters stroke (that we referred to is the regular script, Kai, which we are going to synthesize here), the most famous one is the eight-laws of character 'yong' (永). It is the basic rule of writing in Chinese calligraphy and is extremely important for beginners. People who want to write calligraphic characters well should learn it first.

The eight-laws of character ‘yong’ contains: (1) spot (點,側), (2) horizontal stroke (橫,勒), (3) vertical stroke (豎,努), (4) rise suddenly (鈎,趯), (5) horizontally rise stroke (仰橫,策), (6) long left-falling stroke (長撇,掠), (7) short left-falling stroke (短撇,啄), and (8) right-falling stroke (捺,磔). Figure 4.4 shows it.

Figure 4.4 The eight-laws of character ‘yong’ (永字八法).

We apply the concept of the eight-laws of character ‘yong’ into the calligraphy system. For the purpose of imitating the calligraphy written by Yan, we made some revision on the eight-laws and rearrange the strokes of Chinese characters following the taxonomy of [1]. By the classifying rules, the stroke categories are modified to (A) horizontal stroke (橫), (B) vertical stroke (豎), (C) left-falling stroke (撇), (D) right-falling stroke (捺), (E) spot (點), (F) rise (挑), (G) arc (厥), and (H) long-arc (鈎). And these eight kinds of stroke all have eight kinds of writing style, respectively.

At the same time, in order to let the strokes more suitable for our system, some of the writing styles of certain stroke are added, and some are deleted. We also simplify all strokes to one direction since the system load can be lowered in this way (with the concept of the eight-law of the character 'yong'). The final version of the stroke set is showed in Figure 4.5.

4.2.1 Naming

According to the stroke types, we can design each stroke style by setting the parameters of stroke profile with the stroke model discussed in Section 4.1. As noted previously, the purpose of the stroke type classification is offering choice as the inference result in the system. So we should name the strokes properly. The naming rules comprise three components, which are the main type classification, secondary type classification, and the inference classification. We can see more clearly in Section 4.3. For example, "A1a" stands for a stroke in which 'A' means the horizontal stroke, the first type in stroke type classification; '1' means the 1st kind of 'A' type; and 'a' indicate the choice 'a' of 'A1' type after the inference of expert system. The naming is also shown in Figure 4.5.

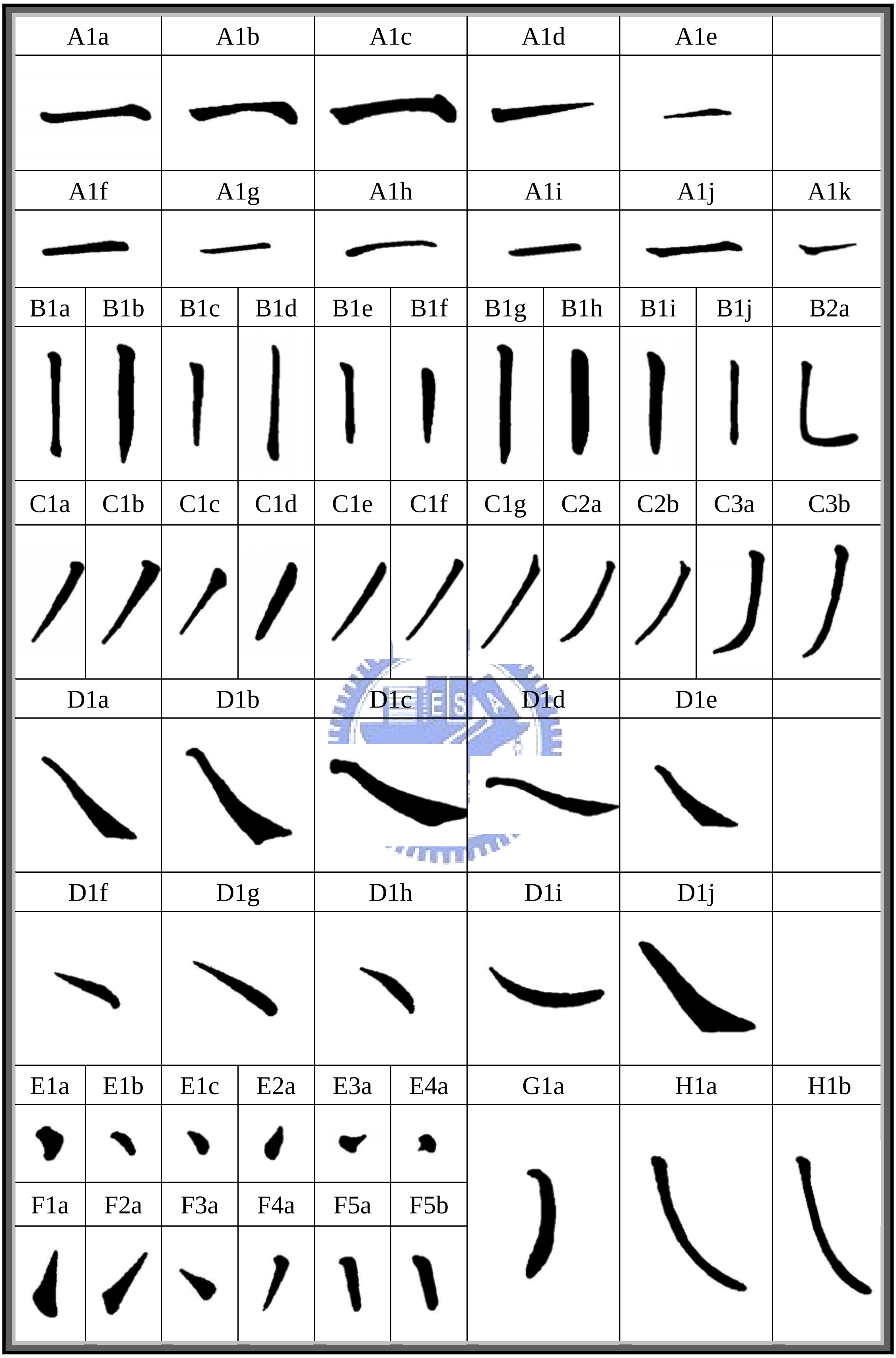

Figure 4.5 The name and classification of strokes for Yan's style.

4.3 System Progress: Two Phases

When stroke model and classification of stroke types both are ready, we can enter the main decision mechanism in the system. There are two phases in the progress. The first phase is the classification by direction. In this part, system will make the preliminary categorization. This includes the main type classification and the secondary type classification that has already been discussed in Section 4.2.1. Through this phase, stroke will get the first two symbols of its name, for example, A1 or E1. This phase is executed in the C++ environment and it belongs to the first module in the system structure as shown in Figure 4.1.

Then in the second phase, we will choose more suitable style for every kind of stroke. Via the inference of the expert system, stroke will get the last symbol of its name. This stroke appears with the description of its profile finally. This phase is executed in the JAVA environment and it belongs to both the second and the third module in the system structure as shown in Figure 4.1.

4.3.1 Phase 1: Classify by Direction

When a stroke is written (it means the trajectory not been dealt with yet), it contains a lot of information immediately, such as the position on which the virtual pen touches the paper, the length of the stroke, and the direction of the stroke, etc. Among them, the obvious one is the direction. It dominates which type the current stroke belongs to. The direction can be obtained via the stroke slope and the relative relation of the start and end points in a stroke.

Figure 4.6 shows the direction graph describing the length and the slope of strokes. Because the paper size in our system is 400 x 400 pixels, each stroke ranges from 0 pixels to $400\sqrt{2}$ pixels. The center point in Figure 4.6 stands for the start point of one stroke. With the direction of it, the stroke ends in one range of certain stroke type. From now, we can classify strokes by this graph.

The classification can proceed from inside to outside. First we compare the current stroke and the strokes with short length, is that, E1, E2, E3, and E4, which are independent to each other and the length of them are about 40 to 60 pixels at most. The 'independent' here means the range of them doesn't overlap those of the others. In more detail, the range of E1 is 0 to 60 pixels with slope from –3 to –1, and 0 to 80 pixels with slope from –1 to –0.1. The range of E2 is 0 to 60 pixels with slope from 2 to ∞ and from $-\infty$ to –3. E3 and E4 can be described in the same way.

If this stroke does not match these types above, we compare it with the strokes that have the longer length about 60 to 120 pixels at most, such as F1, F2, F3, F4, and F5. If the current stroke still not matches those ones, it will be compared with the longest ones about 400 or more pixels at most that are C1, C2, C3, G1, B1, B2, H1, D1, and A1 finally.

Another important decision factor can be used to categorize the stroke, which is the deviation of it. We can see that the ranges of C1, C2 and C3 overlap to each other in Figure 4.6. The same situation also takes place in B1 and G1, H1 and B2, and D1 and B2. The stroke cannot be classified just with the slope between the starting and ending points of it here. But with the stroke deviation, we can achieve this goal. For example, the comparison of deviation among C1 and C2 and C3 is C1<C2<C3; the comparison of deviation between B1 and G1 is B1<G1; the shape of stroke B2 is like a right angle but that of stroke D1 and H1 are not (that the shape also influence the deviation). Finally, the directional classification is done and each stroke has the first two symbols of its name now.

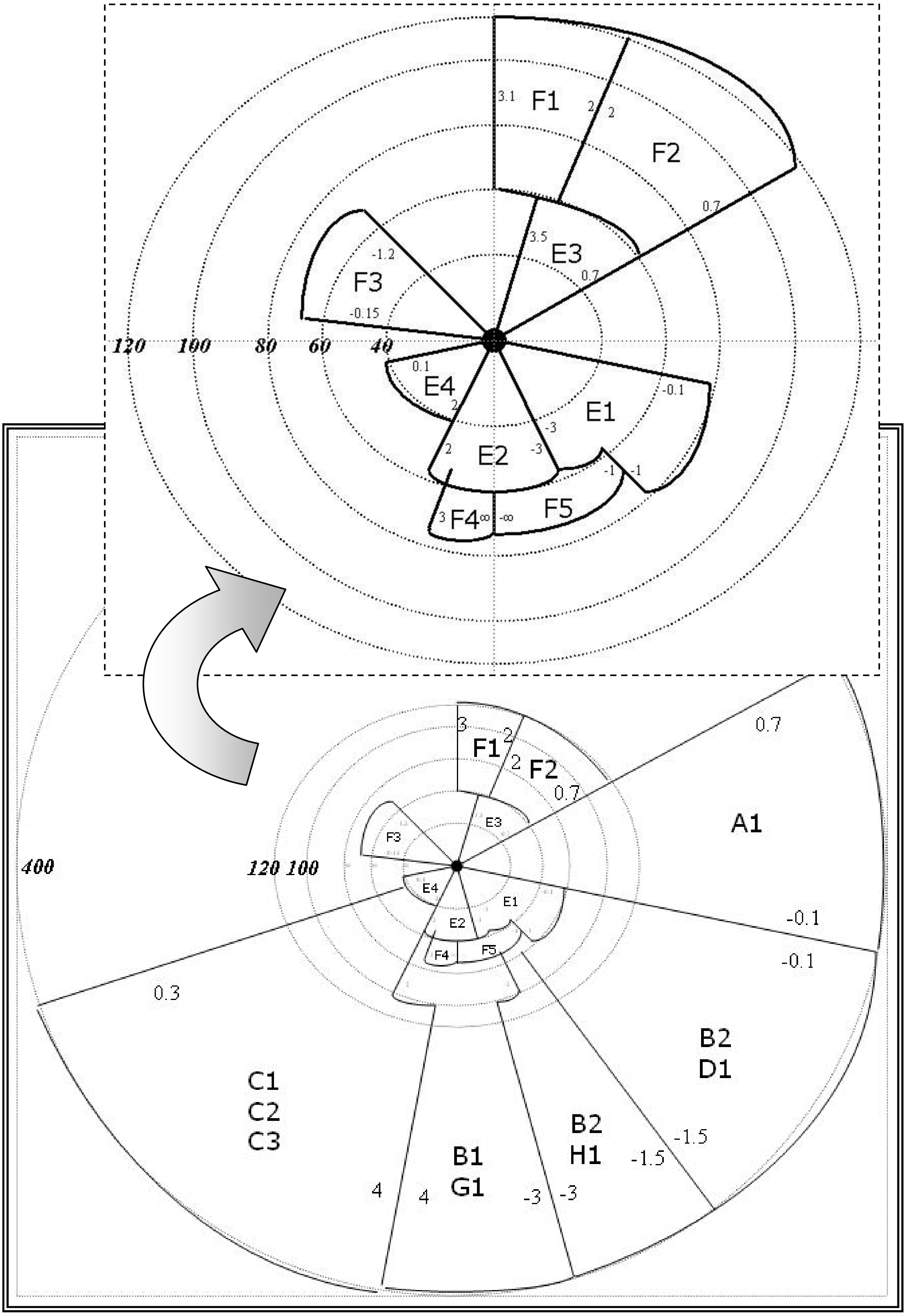

Figure 4.6 Direction graph with length (in pixels) and slope for stroke classification in the first phase for Yan's style.

4.3.2 Phase 2: Classify by Relationship

After the directional classification, the first two symbols in the name of the current stroke can be determined. Then, getting the third symbol would be taken as our goal that can be done by the relationship between the current stroke and the other strokes in a character. There are two components in this phase, including the definition of attribute set and rules designed.

4.3.2.1 Attribute Set

As stated before, when a stroke is written, it gets a lot of information immediately. We used the direction and length as the criterion to determine which kind of type a stroke belongs to. Here, more information of one stroke will be gathered and used to do inference in expert system.

We define the attribute set for each stroke. It has five subsets: The first one states the input information before entering inference engine; The second subset is the information about stroke coordinate, including the x/y coordinates of the starting and ending points; The third subset is the global information about the current character; The fourth subset is the local information about every stroke; And the last one is the information of relationship between strokes. The more detailed descriptions are enumerated in Appendix.

4.3.2.2 Rules Design

Rules design is the most central part in the whole calligraphy system. It is the third module in the system structure as shown in figure 4.1. The quality of it will influence the appearing of the last result. We use DRAMA as our development platform to design complete and perfect rules.

Rules design for calligraphy follows some principles:

1. **Classify the stroke by the number of strokes in a character.** In most cases, stroke in a character with larger number of strokes will appear in a narrower width. On the contrary, the one with less number of strokes will appear in a wider width. Hence, the number of strokes in a character is considered importantly.

2. **Classify the stroke by the start and end position.** Except for the usage in the classification by direction, the start and end position are also important for inference. By the concept of "nine-grid" (九宮格) in Chinese calligraphy, the location of the stroke head and stroke tail can be analyzed definitely. So the stroke style will be determined appropriately.

3. **Classify the stroke by its length.** The different stroke length in the same stroke type may cause different styles. For example, the stroke in type

"A1" can be assigned the style "A1b" for the longer length, to show the main part of the character. However, "A1g" may be the better style for the shorter stroke, which is like a supporting role in a word. The example is shown in Figure 4.7 (a).

4. **Classify the stroke by its stroke-number order in the word.** For example, if the stroke "spot" (E1) is written at the first stroke-number order, it often demonstrates the imposing manner at the beginning of a character as style "E1a". If it is not the first one, it may be comparatively appropriate to behave with style "E1b". Figure 4.7 (b) shows the difference between the two styles.

5. **Classify the stroke by the order among strokes.** The most common example is the radical "衤". In contains the strokes "A1", "A1", "C2", "B1" and "E1" in order. By this specific regular order, the style for each stroke can be determined easily as shown in Figure 4.7 (c).

6. **Classify the stroke by the overlap among strokes.** If the left side is not overlapped with another stroke but the right side is, the stroke "A1" probably appear in style "A1d"; on the contrary, it may be appear in style "A1e". This situation is shown in Figure 4.7 (d).

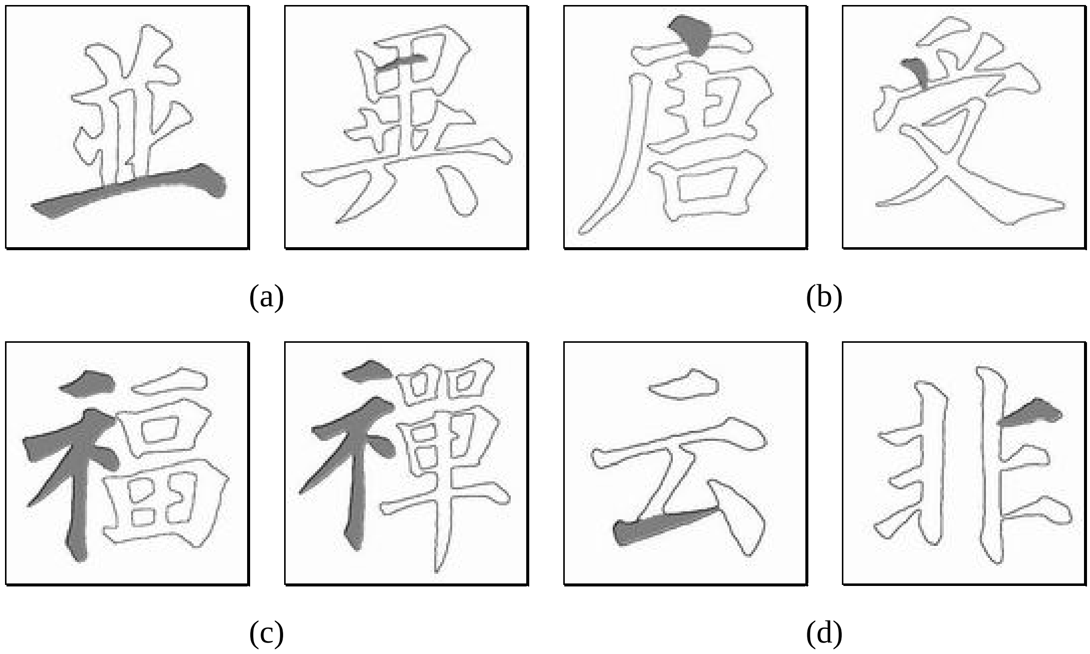

Figure 4.7 Examples for the principles of rule design. (a) length, (b) stroke-number order, (c) strokes-order relation, and (d) stroke overlap.

In rules design, we can follow the principles singly, or follow them in a compound way. In other words, the rules can be designed with two or more principles at the same time. For example, if the right-falling stroke begins at the left-lower grid (in Chinese "nine-grid") and ends at the right-lower grid, its style may be "D1c" or "D1d" according to principle 2 (describing the start or end position of the stroke). If we consider principle 1 simultaneously, the right-falling stroke in a character with more stroke number is more suited with style "D1d" that is narrower in width but not "D1c" that is wider in width. We can see the difference in Figure 4.8.

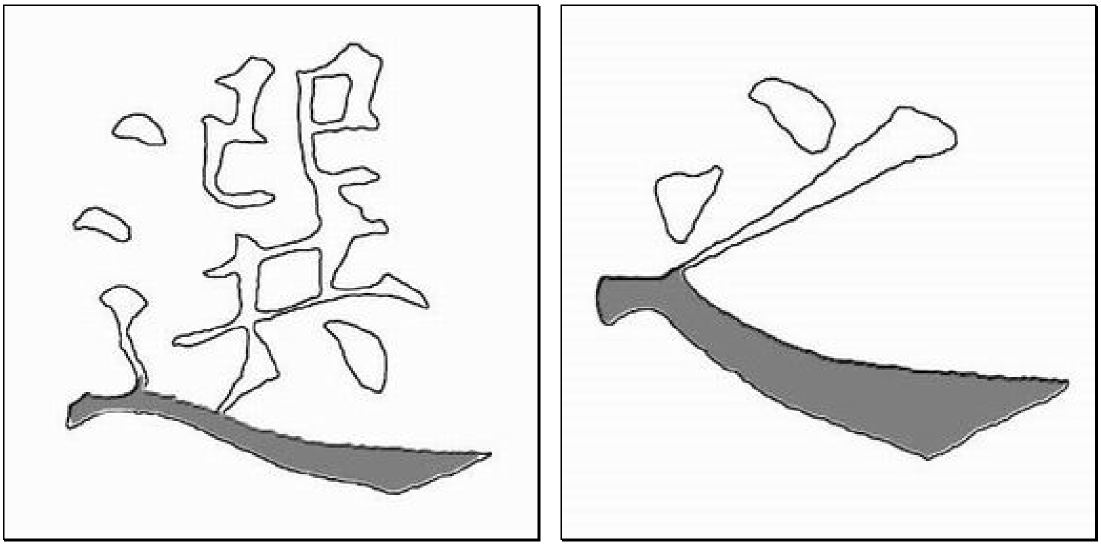

Figure 4.8 Example for the compound principle about stroke position and number of strokes.

With the compound of principles, rules can be made up freely. For example, the pattern "口" often appears in a lot of characters. It comprises four strokes that are "B1", "A1", "B1" and "A1" in order. In Yan's style, sometimes the starting point of the first stroke "B1" is overlapped with another stroke, but sometimes it is not. We do not define this characteristic. But the ending point of it is overlapped with the start position of the fourth stroke "A1". Besides, the pattern "口" makes the first stroke "B1" followed by the strokes "A1" and "B1". The order is also considered in this situation. By this way, these four strokes may be matched with the styles "B1e", "A1g", "B1f" and "A1f".

Finally, all rules for inference in calligraphy system are set completely in the same way. We will reserve these rules in Appendix in detail.

Each stroke is assigned a specific style after two main classifications (by direction and by relation). With the stroke model mentioned in the beginning of this chapter, every stroke can be drawn clearly on the paper. All strokes are combined to form a character. According to rules made for Yan, this character will be generated in Yan's style at last. And the work of the calligraphy system is done.

Chapter 5

Experimental Results

The implemental results of our proposed system are presented in this chapter. The system is written in both C++ language with OpenGL API and JAVA language with JAVA-based DRAMA API on the PC platform with an AMD 1.4GHz CPU and 256 MB RAM.

The system provides a convenient user interface as shown in Figure 5.1. The upper left part is the DRAMA server which have full knowledge base and provide service for intelligent decisions. The upper right part is the inference engine which handle the whole inference procedure. Below are three calligraphy I/O. The left one shows the original copybook of calligraphy written by Yan. The right one shows the trajectories of all strokes we wrote by imitating the left one. The middle one shows the final result with all stroke styles representing the stroke trajectories.

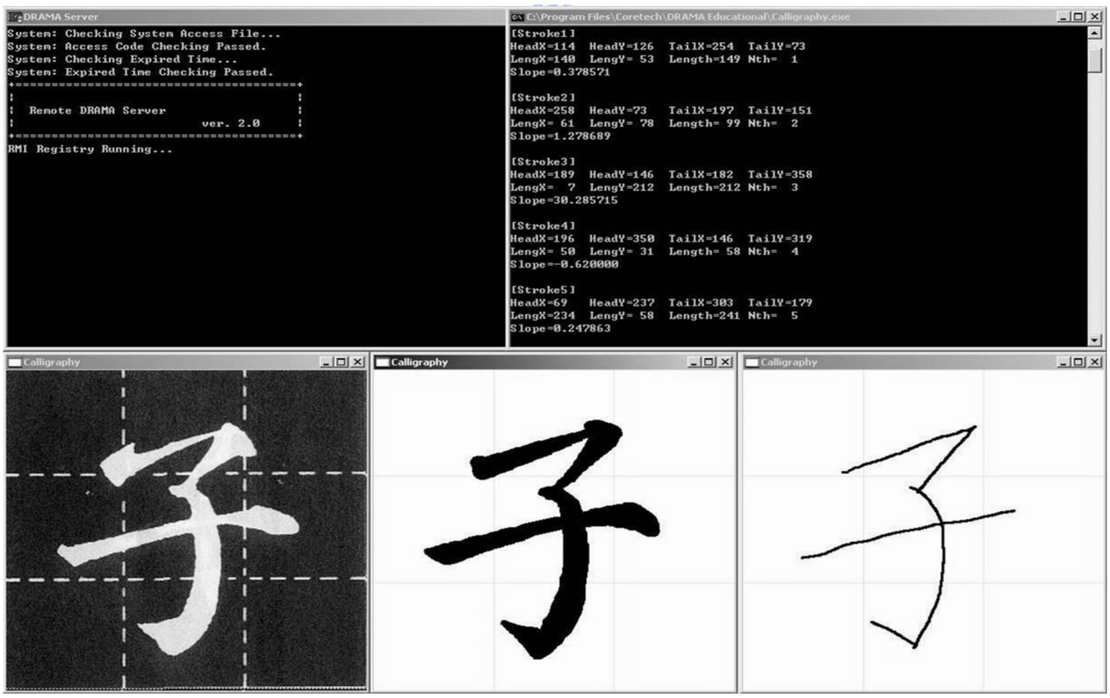

Figure 5.1 User interface. The upper left part is DRAMA server and upper right part is inference engine. Below are three I/O for original copybook (left) and the result we generate (middle) and the corresponding trajectories (right). The demonstrating example here is an Chinese character ‘子’.

Figure 5.2 shows the synthesis of the character ‘yong’ (永). Left is the reference copybook of it. Right is the generated result of our system and we attach the corresponding stroke styles on it.

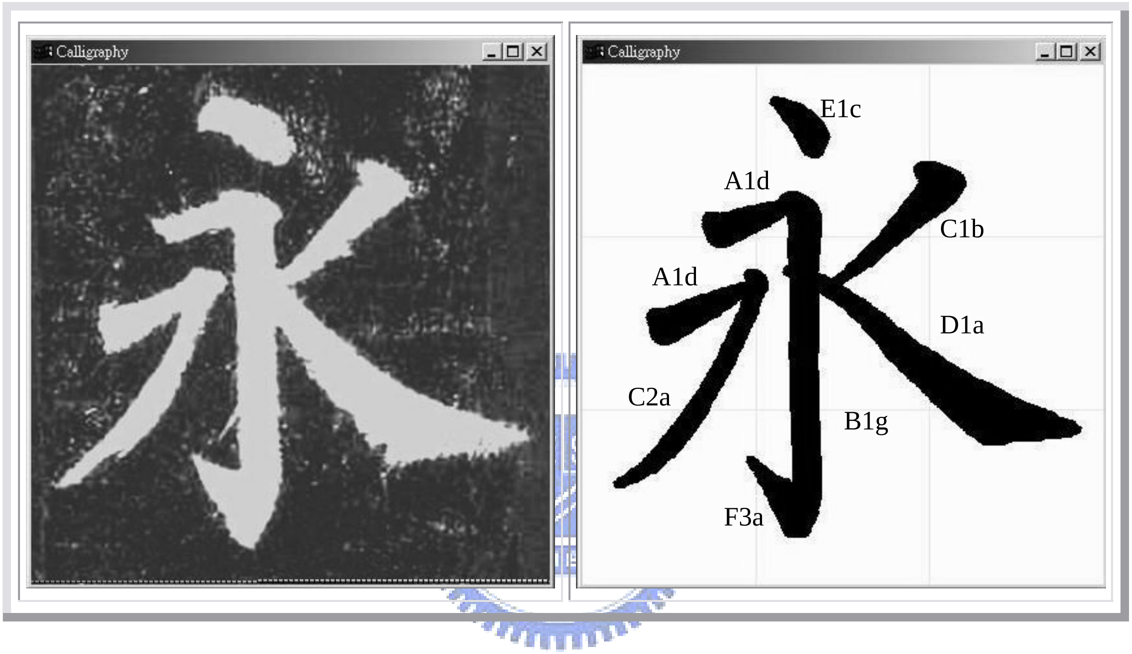

Figure 5.2 The synthesis of character ‘yong’ (永) with all stroke styles.

Figure 5.3 shows the synthesis of several characters and the corresponding trajectories with stroke styles.

Figure 5.3 Synthesis of several characters in [1].

Figure 5.4 shows synthesis of other characters. Left is the original copybook of them. Right is the generated results from our calligraphy system.

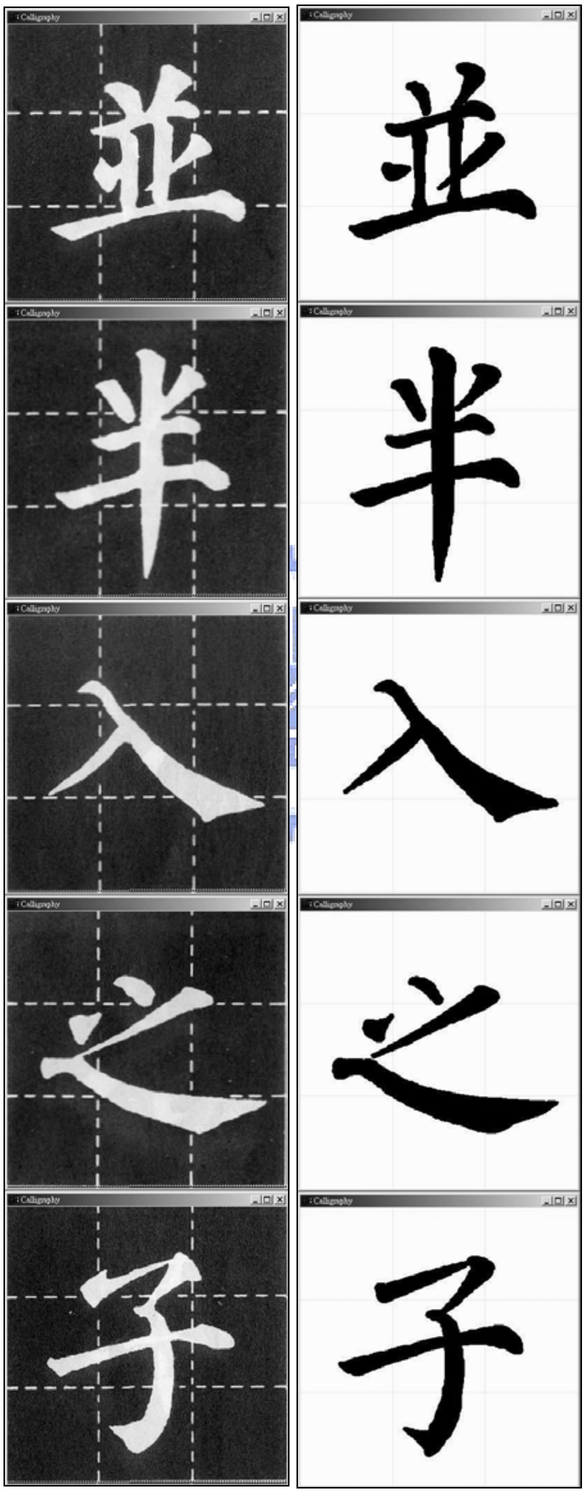

Figure 5.4 Synthesis of several characters in [1].

Figure 5.5 shows the synthesis of one-page copybook which contains about 15 characters with some stamps. Left is the original one. Right is the generated result.

Figure 5.5 Synthesis of one-page calligraphic copybook [3].

Figure 5.6 shows another result of one-page synthesis of calligraphic copybook.

Figure 5.6 Synthesis of another one-page calligraphic copybook [3].

Chapter 6

Conclusions & Future Works

In this thesis, we propose a knowledge-based system for Chinese calligraphy synthesis. Users can generate calligraphic characters in Yan's style by imitating his calligraphic copybook, which are completed by tracing the stroke trajectories only and users don't have to be familiar with the Chinese calligraphy art.

There are five modules in our proposed system which are considered as the system procedures in order. The first one is stroke analysis for analyzing every stroke to classify the stroke type. The second module determines the relationships among all strokes and sets attributes of them. Then, stroke styles are determined in the third module, which uses an expert system, DRAMA, as inference engine. In the fourth module, system collects the stroke style once a time. Finally, the fifth module forms a calligraphic character by all the strokes with their specific styles, and the character will appear in Yan's Kai style.

The main feature of this calligraphy system is the integration of computer graphics and knowledge engineering. With computer graphics, the graphical behavior of calligraphy can be presented in the virtual world. With knowledge engineering, the aesthetic calligraphic characters in Yan’s style can be generated intelligently. We combine these two different fields in computer science to achieve better results.

However, there are several topics for further research:

1. Our system focuses on synthesizing Yan’s Kai style. The expansion of this system can be done by imitating other famous calligraphers’ styles, such as Liu Gong-Quan(柳公權) and Qi Bai-Shi(齊白石). Modifying the stroke profiles and inference rules can achieve this goal.
2. Although the classification method is fine, wrong decisions are made sometimes. The influence factors include the size of copybook, the clarity of copybook, and the writing styles of calligraphers in different mood. We should handle these exceptions to increase system accuracy.
3. The proposed system is a semi-automatic one. Users generate calligraphic characters in Yan’s style by tracking stroke trajectories of his calligraphic copybook for imitating. Further more, we hope to improve the system to let users write characters freely, not limited to the trajectories. This work can be done by modifying the whole rule structure.

Appendix –

Inference Rules in Grid

The attribute set has 5 subsets:

1. The first set contains just one attribute. ***INPUT***: the first two symbols of the stroke name.
2. The second subset is the information about stroke coordinate. ***HEADX***, ***HEADY***, ***TAILX*** and ***TAILY***: representing the x coordinate of the start point, y coordinate of the start point, x coordinate of the end point and y coordinate of the end point in one stroke, respectively.
3. The third subset is the global information about the current character. ***STRKN*** represents the total number of strokes in this character. ***MAXX*** is the length in the direction parallel to x-axis that is the longest one among all strokes. ***MAXY*** is the length in the direction parallel to y-axis that is the longest one among all strokes.
4. The fourth subset is the local information about every stroke. ***NTH*** means the writing order of the stroke in the character. ***LENGX*** means the length in x-direction. ***LENGY*** means the length in y-direction.
5. The fifth subset is the information of relationship between strokes. ***HEAD*** and

TAIL state which stroke the current stroke touches, and in what part it touches. ***BEFORE1***, ***NEXT1***, ***BEFORE2***, and ***NEXT2*** representing the last stroke, the next stroke, the stroke before the last one, and the stroke next to the next one, respectively.

INPUT	*HEADX*	*HEADY*	*TAILX*	*TAILY*	*STRKN*	*MAXX*	*MAXY*	*NTH*	*LENGX*	*LENGY*	*HEAD*	*TAIL*	*BEFORE1*	*NEXT1*	*BEFORE2*	*NEXT2*	**STYLE**
A1	-	-	-	-	-	-	-	-	>=100 <150	-	NULL	NULL	-	-	-	-	A1a
A1	-	-	-	-	>15	-	-	-	>=150	-	NULL	NULL	-	-	-	-	A1a
A1	-	-	-	-	>=4 <15	-	-	-	>150	-	NULL	NULL	-	-	-	-	A1b
A1	-	-	-	-	<4	-	-	-	>200	-	NULL	NULL	-	-	-	-	A1c
A1	-	-	-	-	-	-	-	-	-	-	NULL	B1B or B1T or C1H or C2H or C1B or	-	-	-	-	A1d
A1	-	-	-	-	-	-	-	-	-	-	C1T	NULL or E1B	-	-	-	-	A1d
A1	-	-	-	-	>5	-	-	-	-	-	B1H or B1B or B1T	NULL	-	-	-	-	A1e
A1	-	-	-	-	-	-	-	-	-	-	B1T or F5T	B1T	-	-	-	-	A1f

INPUT	*HEADX*	*HEADY*	*TAILX*	*TAILY*	*STRKN*	*MAXX*	*MAXY*	*NTH*	*LENGX*	*LENGY*	*HEAD*	*TAIL*	*BEFORE1*	*NEXT1*	*BEFORE2*	*NEXT2*	**STYLE**
A1	--	>=100	--	--	>=5 <12	--	--	--	>=50 <100	--	NULL	NULL	--	--	--	--	A1f
A1	--	--	--	--	>=15	--	--	--	--	--	--	B1H	B1 or F5	B1	--	--	A1g
A1	--	--	--	--	--	--	--	--	<50	--	--	--	--	--	--	--	A1g
A1	--	--	--	--	--	--	--	--	--	--	--	--	A1	A1	--	--	A1h
A1	--	<100	--	--	--	--	--	--	>=50 <100	--	NULL	NULL	--	--	--	--	A1i
A1	--	>=100	--	--	>=12	--	--	--	>=50 <100	--	NULL	NULL	--	--	--	--	A1i
A1	--	--	--	--	<15	--	--	--	--	--		B1H	B1 or F5	B1	--	--	A1j
A1	--	--	--	--	--	--	--	--	--	--	B1H or B1T	--	--	--	--	--	A1j
A1	--	--	--	--	--	--	--	--	--	--	--	--	C1	C1	--	--	A1k
B1	>=250 or <150	--	--	--	--	--	--	!STRKN	--	>=200	--	--	!C1 !C2	!F3	--	--	B1a

B1	B1	B1	B1	B1	B1	B1	B1	B1	B1	**INPUT**
--	--	--	--	--	--	--	>=150 <250	--	--	*HEADX*
--	--	--	--	--	--	--	--	--	--	*HEADY*
--	--	--	--	--	--	--	--	--	--	*TAILX*
--	--	--	--	--	--	--	--	--	--	*TAILY*
--	--	--	--	--	--	<10	--	--	--	*STRKN*
--	--	--	--	--	--	--	--	--	--	*MAXX*
--	--	--	--	--	--	--	--	--	--	*MAXY*
=STRKN	--	--	--	--	--	=STRKN	--	--	!STRKN	*NTH*
--	--	--	--	--	--	--	--	--	--	*LENGX*
>=133 <200	--	<120	--	>=133	<133	>=200	>=200	<133	>=133 <200	*LENGY*
--	--	A1T	--	--	NULL	--	--	NULL	!A1T	*HEAD*
--	--	A1T or A1B	A1H or A1B	--	!NULL !A1H	--	--	NULL	!A1T	*TAIL*
--	--	A1	--	C1 or C2	--	--	--	--	!C1 !C2	*BEFORE1*
--	F3	--	A1	--	--	--	!F3	--	!F3	*NEXT1*
--	--	--	--	--	--	--	--	--	--	*BEFORE2*
--	--	--	B1	--	!B1	--	--	--	--	*NEXT2*
B1g	B1g	B1f	B1e	B1d	B1c	B1b	B1b	B1a	B1a	**STYLE**

INPUT	B1	B1	B1	B1	C1	C1	C1	C1	C1	C1
HEADX	--	--	--	--	--	<200	--	--	--	--
HEADY	--	--	--	--	--	--	--	--	--	--
TAILX	--	--	--	--	--	--	--	--	--	--
TAILY	--	--	--	--	>=230	--	<230	--	--	--
STRKN	>10	--	--	--	<12	>=12	<12	>=12	>=12	>=12
MAXX	--	--	--	--	--	--	--	--	--	--
MAXY	--	--	--	--	--	--	--	--	--	--
NTH	=STRKN	--	--	--	--	--	--	--	--	--
LENGX	--	--	--	--	--	--	--	--	--	--
LENGY	>=200	<133	>=120	--	--	--	--	--	--	--
HEAD	--	!NULL !A1T	A1T	--	--	--	--	--	--	--
TAIL	--	!NULL !A1H !A1T	A1T	--	--	--	--	--	--	--
BEFORE1	--	--	--	A1	--	E1	!A1	--	--	--
NEXT1	--	--	--	A1	--	A1	!A1	B1	!A1	A1
BEFORE2	--	--	--	--	--	--	--	--	--	--
NEXT2	--	--	--	--	--	--	--	E1	!B1	B1
STYLE	B1g	B1h	B1i	B1j	C1a	C1a	C1b	C1c	C1d	C1e

INPUT	C1	C1	C1	C2	C3	C3	D1	D1	D1	D1
HEADX	--	--	>=200	--	--	--	--	--	--	--
HEADY	--	--	--	--	--	--	--	<200	>=200	>=200
TAILX	--	--	--	--	--	--	--	--	--	--
TAILY	<230	>230	--	--	--	--	--	--	--	--
STRKN	<12	<12	>=12	--	<5	>=5	>=5	--	<10	>=10
MAXX	--	--	--	--	--	--	--	--	--	--
MAXY	--	--	--	--	--	--	--	--	--	--
NTH	--	--	--	--	--	--	=STRKN	=STRKN	=STRKN	=STRKN
LENGX	--	--	--	--	--	--	>=155	--	--	--
LENGY	--	--	--	--	--	--	--	--	--	--
HEAD	--	--	--	--	--	--	!NULL	NULL	NULL	NULL
TAIL	--	--	--	--	--	--	--	--	--	--
BEFORE1	A1	A1	E1	--	--	--	--	--	--	--
NEXT1	A1	F3	A1	--	--	--	--	--	--	--
BEFORE2	--	--	--	--	--	--	--	--	--	--
NEXT2	--	--	--	--	--	--	--	--	--	--
STYLE	C1f	C1f	C1g	C2a	C3a	C3b	D1a	D1b	D1c	D1d

E1	E1	E1	E1	D1	D1	D1	D1	D1	D1	**INPUT**
--	--	--	--	--	--	--	--	--	--	*HEADX*
--	--	--	--	--	--	--	--	--	--	*HEADY*
--	--	--	--	--	--	--	--	--	--	*TAILX*
--	--	--	--	--	--	--	--	--	--	*TAILY*
--	<10	--	>=10	<5	--	--	--	--	--	*STRKN*
--	--	--	--	--	--	--	--	--	--	*MAXX*
--	--	--	--	--	--	--	--	--	--	*MAXY*
=STRKN	=1	!=1 !=STRKN	=1	=STRKN	!=STRKN	!=STRKN	!=STRKN	!=STRKN	=STRKN	*NTH*
--	--	--	--	>=155	--	--	--	--	<155	*LENGX*
--	--	--	--	--	--	--	--	--	--	*LENGY*
--	--	--	--	!NULL	--	--	C1T or C2T	!C1T !C2T	!NULL	*HEAD*
--	--	--	--	--	--	--	--	--	--	*TAIL*
--	--	--	--	--	E2	C1	--	--	--	*BEFORE1*
--	--	--	--	--	F3	F3	!F3	!F3	--	*NEXT1*
--	--	--	--	--	--	--	--	--	--	*BEFORE2*
--	--	--	--	--	--	--	--	--	--	*NEXT2*
E1c	E1c	E1b	E1a	D1j	D1i	D1h	D1g	D1f	D1e	**STYLE**

H1	H1	G1	F5	F4	F3	F2	F1	E4	E3	E2	B2	**INPUT**
--	--	--	--	--	--	--	--	--	--	--	--	*HEADX*
--	--	--	--	--	--	--	--	--	--	--	--	*HEADY*
--	--	--	--	--	--	--	--	--	--	--	--	*TAILX*
--	--	--	--	--	--	--	--	--	--	--	--	*TAILY*
>=10	<10	--	--	--	--	--	--	--	--	--	--	*STRKN*
--	--	--	--	--	--	--	--	--	--	--	--	*MAXX*
--	--	--	--	--	--	--	--	--	--	--	--	*MAXY*
--	--	--	--	--	--	--	--	--	--	--	--	*NTH*
--	--	--	--	--	--	--	--	--	--	--	--	*LENGX*
--	--	--	--	--	--	--	--	--	--	--	--	*LENGY*
--	--	--	--	--	--	--	--	--	--	--	--	*HEAD*
--	--	--	--	--	--	--	--	--	--	--	--	*TAIL*
--	--	--	--	--	--	--	--	--	--	--	--	*BEFORE1*
--	--	--	--	--	--	--	--	--	--	--	--	*NEXT1*
--	--	--	--	--	--	--	--	--	--	--	--	*BEFORE2*
--	--	--	--	--	--	--	--	--	--	--	--	*NEXT2*
H1b	H1a	G1a	F5a	F4a	F3a	F2a	F1a	E4a	E3a	E2a	B2a	**STYLE**

Reference

[1] 何睿晃、胡凡，顏真卿楷書寫法，藝術圖書公司出版。

[2] 松丸道雄，玄秘塔碑–唐柳公權，株式会社二玄社出版。

[3] 松丸道雄，多寶塔碑–唐顏真卿，株式会社二玄社出版。

[4] 曾憲雄、黃國禎、江孟峰、蔡昌均、林耀聰，專家系統導論工具應用，文魁出版。

[5] 閔祥德、廖蘊玉，書法百問百答–楷書書寫門徑，莊嚴出版社出版。

[6] 趙英山，書法新義，商務公司出版。

[7] Booch G., Object-oriented design with applications, 2nd edition, San Francisco, CA: Benjamin/Cummings, 1991.

[8] Buchanan B.G.., Shortliffe E.H., Rule-based expert systems, *London:* Adison-Wesley, 1984.

[9] Helena T.F. Wong, Horace H. S. Ip, “Virtual brush: a model-based synthesis of Chinese calligraphy”, Computer & graphics 2000, pp. 99-113.

[10] Horace H S Ip, Helena T F Wong, “Calligraphic character synthesis using a brush model”, *Proceedings of Computer Graphics International 1997*, pp. 13-21.

[11] Mi Xiaofeng, Xu Jie, Tang Min, Dong Jinxiang, “The droplet virtual brush for Chinese calligraphic character modeling”, *Proceedings of the sixth IEEE workshop on applications of computer vision 2002*, pp. 330-334.

[12] Nelson S. –H. Chu, Chiew-Lan Tai, “An efficient brush model for physically-based 3D paingting”, *Proceedings of the 10th Pacific Conference on CG&A.*

[13] Porch KC, Fellner WD, “The circle-brush algorithm.”, *ACM Transactions on Graphics*, Vol. 8, No. 1, pp. 1-24, January 1989.

[14] Roesner H., “Expert systems for commercial use”, *Artificial intelligence and expert systems 1988*, pp. 35-59.

[15] Rumbaugh J., Blana M., et al., Object-oriented modeling and design, Prentice Hall, 1991.

[16] Strassmann S, “Hairy brushes”, *Computer Graphics*, Vol. 20, pp. 225-232, August 1986.

[17] Wei Li, Ichiro Hagiwara, Takao Yasui, Hu-awei Chen, “A method of generating scratched look calligraphy characters using mathematical morphology”, *Joural of computational and applied mathematics 2003*, pp. 85-90.

[18] Yang His-Ming, Lu Jainn-Jyh, Lee His-Jian, “A Bezier curve-based approach to shape description for Chinese calligraphy characters”, *International conference on document analysis and recognition 2001*, Sec. 1, pp. 276-280.