Dynamic Glyph Generation Based on variable length encoding schema

Yap Cheah Shen [1)]

Abstract

About 20 years ago, Prof. Hsieh Ching-Chun from Academia Sinica proposed a descriptive method for encoding **Hanzi** (Kanji), as a fundamental guideline to deal with the missing Chinese character problem. As we (eforth Techology, Inc) are trying to develop an OS for our embedded CPU, we have a chance to design CJK environment from scratch. We soon found out the key part of the system is a module which takes glyph expression as input, and generates glyph outline as output. It is called dynamic glyph generation. The term "dynamic" emphasizes glyph outline is generated on the fly, in contrast to ordinary static font file approach.

1 Morpheme: Latin vs. Han

A Morpheme is the smallest meaningful unit in the grammar of a language. In Indo-European languages, it is "word" separated by space[1]. In Chinese, there are no spaces, the morpheme is Hanzi.

Morpheme is the basic unit which map to the real world ideas. No doubt the set of morphemes will keep changing and expanding from time to time, from location to location. "Changing" means a word maps to different[2] ideas, and "expanding" means new[3] word is created.

2 Latin text encoding

Numerous types of protein molecule are constructed from only 20 amino acids. Likewise, all English words can be represented by sequence of alphabets. The natural encoding unit for Indo-European languages is alphabet.This is how ASCII works, encoding alphabets and symbols, not "words".

To encode words as sequences of alphabets is neither space nor time efficient. Fixed-bit or Huffman encoding is more compact and requires less computing effort. We choose the less efficient way because of extensibility, to ensure the encoding system remain unchanged, no matter how many new words are introduced.

3 Missing Characters in Chinese Text

Hanzi are in an open-set, theoretically, historically and practically. Therefore, **Hanzi** can never be totally encoded with fix-byte, existing encoding schema, including Unicode, which does not respect the nature of Chinese language and made a wrong approach.

The consequence is bound to be miserable. Unending loop of assigning code point to characters, waiting for OS vendor to release patches or updates, waiting for new input method table, buying new fonts, etc. The users suffer in every stage. But, vendors are happy by generating profit out of the process.

Surely this is not an ideal way of dealing with CJK texts. Because of the wrong design, users and the society keep suffering and paying high cost. How can we come out of this misery?

4 Solution

Unlike alphabet, the basic unit of **Hanzi** is more subtle. Most **Hanzi**[4] are form by two or more "parts" or "components". The number of component is quite limited, and some have really high use frequency. Prof. Hsieh has done a lot of work[5] on this. Although most components are **Hanzi** themselves, but it is not necessary so. Components show irregular distribution. A small number of components occur quite frequently, but many components

[1)] eForth Techology. Inc.

[1] To be more precise, "unladylike" is a word, but consisting of 3 morpheme: un- , lady and –like. We don't have to worry about the linguistic definition in our discussions.

[2] For example, "current" maps to ideas of "con-temporary" and "flow of water". Later, it refers to "flow of electricity".

[3] Brands, Abbreviations, etc.

[4] More than 85% our of 70000 Hanzi are composite.

[5] Please visit `http://www.sinica.edu.tw/~cdp/paper/1996/19961005_1.htm`

only happen once or twice. Chances are rare that new components are needed for new **Hanzi**. To simplify the system, we pick about 800 components as a basic set. Basic strokes are used for describing components which are not included.

We come out with a closed set of basic components and strokes as encoding units.

Because **Hanzi** is two dimensional, we need three operators for describing how component are joined: horizontal, vertical, and enclosing. We add a new shielding operator, which hide strokes. It is useful for describing tabooed[6] **Hanzi**. Prefix notation is used, as suggested by Unicode IDS specification.

A CJK environment can be divided into three aspect: inputting, displaying/printing, and storing/interchanging.

In ordinary CJK environment, a fix numeric value is given to each **Hanzi**, which is call the "character code".

Character code is difficult for human being to memorized. The most logical method is to split **Hanzi** into phoneme or components, and map each phoneme and component to a key. **Hanzi** can be represented as sequences of keystrokes. Different input method has different way of mapping. But, the principle is still the same. Inputting of **Hanzi** is a table-checking problem. That's why even there are hundreds of input methods, none of them can input a Hanzi which has no character code.

The glyph of a **Hanzi** must be pre-designed and store in a font file, either in outline or raster format. The glyph for a certain **Hanzi** is accessed by a character code. Again, if a character has no character code, no matter how many font types are installed to your computer, you can never get the glyph. Someone may argue that we can assign missing character in EUDC (end-user defined character) area and use font designing tool to create glyph for the Hanzi. This is not a good solution, because other people don't acknowledge your assignment to the **Hanzi**, and worse if he has different **Hanzi** assign to the same character code. Another problem is the EUDC area is limited, and it is never enough to hold all **Hanzi** and their variants.

In ordinary system, a text file is nothing but sequences of character code. If a **Hanzi** doesn't have character code, it cannot be represented in a text file.

Existing input method and text format are compatible with variable length glyph expression. It is very easy to encode three operators and basic component in existing input method. (In fact, most component-based input methods share the same set of common components.) Texts remain unchanged, because ideographic descriptive sequence (IDS) is no different from normal text stream. We have only one problem left: how to get a glyph which are not in font file? The combination of components is endless, not only all **Hanzi** can be represented by IDS, but we're free to create unlimited number of "new" **Hanzi**, which have not be seen before. Obviously the static font file approach cannot meet this purpose. We need a mechanism to generate glyph from an IDS. This is the idea of Dynamic Glyph Generator.

5 Dynamic Glyph Generator

Dynamic Glyph Generator takes IDS or other descriptive sequence as input, namely "glyph composing expression" proposed by Academia Sinica, and "glyph assembling expression" proposed by CBETA[7].

Dynamic Glyph Generator first decomposes the glyph. The result is the sequence of basic component and their proportion. Basic component is constructed by basic strokes. Strokes are stored as bezier curves. The generator then recursively builds up the full glyph with these information.

The output can be raster bitmap of any pixel format, true-type compatible outline, SVG or any other vector format.

The generator is very similar to a typesetting program, trying to fit one dimensional stroke sequences into a 2 dimensional square.

6 Implementation

The system consists of 3 major parts:
Glyph decomposition database: courtesy of Prof. Hsieh from Academia Sinica, Taiwan `http://www.`

[6] In Chinese tradition, to show respect, name of emperor, saint and father should be written differently, In most cases a stroke is taken out. Which are useful clue for paleology and bibliology.

[7] Chinese Buddhist Electronic Text Association, `http://www.cbeta.org`

sinica.edu.tw/~cdp/. Prof. Hsieh and his associates[8] have been working on it for more than 10 years, it is almost impossible for us to start without their work.

Outline of strokes and components:
Beijing ZhongYi Co. http://www.zhongyicts.com.cn/ is a professional outline font vendor, they provide us very high quality outline data for strokes and components.

The eForth system:
eForth system is a standalone computing environment. We have full control over the CPU, the OS and the GUI, so that new idea can be implement without any compromise. Our goal is putting everything into a single chip. CJK environment is built into hardware. Users get everything when power is on. No installation and configuration is needed, and no more software incompability issue.

7 Integrating into existing OS/GUI

Dynamic Glyph Generator is originally design to fit the requirement of a embedded system. It is possible to integrate Dynamic Glyph Generator into existing systems. Here are things to be done:

1. String manipulation library
    - Function to calculate number of characters
    - Function which return characters width.
    - Pattern matching functions.
2. Graphic sub-system
    - Drawing a text line (e.g. ExtTextOut in windows)
    - Text handling widgets
3. Human interface
    - Awareness of glyph expression for mouse caret
    - Awareness of mouse/keyboard selection and delete/backspace.

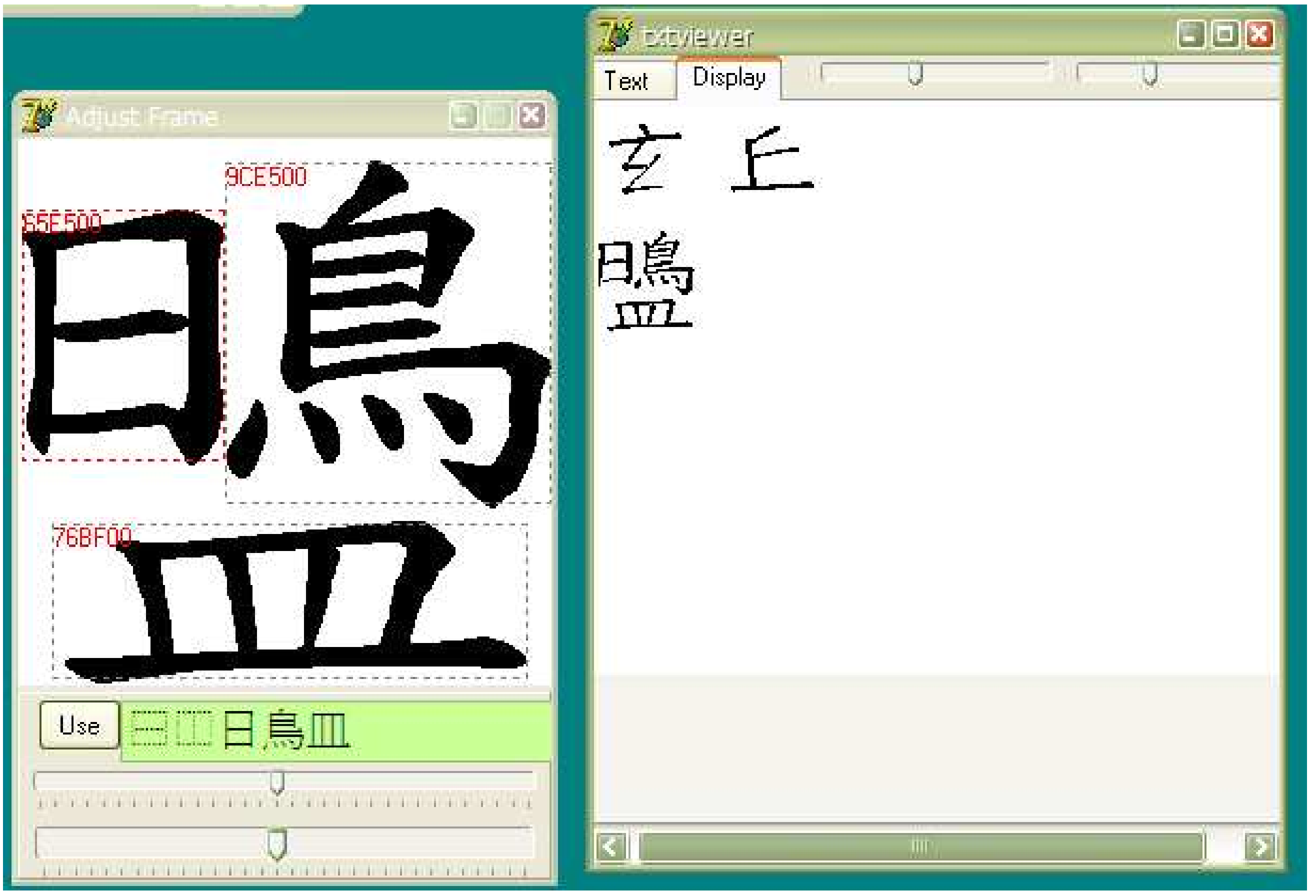
8 Other issues

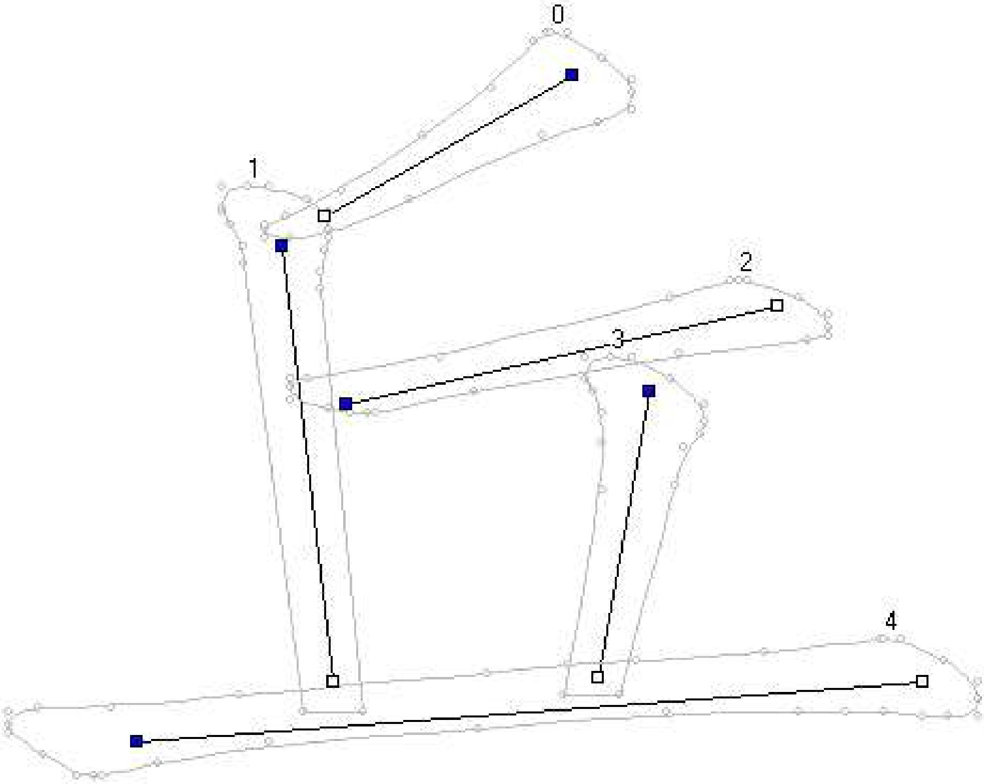
Quality of the glyph:
The quality of the generated glyph can never be as good as pre-designed, fine tuned outline, because the system take only 1/20 ˜ 1/100 space compare to static font approach. Should better quality is needed, common **Hanzi**[9] can be stored as outline, thus offering better quality and increasing overall performance.

Speed of generation:
Glyph generation is never as fast as retrieving glyph from font file, but this is no problem for ordinary system, because glyph generation is rare. For handheld device, we have special hardware acceleration in our CPU, e.g, line drawing and polygon fill, which boost the performance dramatically.

[8] I have to acknowledge Mr.Zhuang Derming, who maintains the database.

[9] Study shows that 1000 Hanzi cover ¿90% of normal text.


Adjust Frame
9CE500
65E500
76BF00
Use
日鳥皿
txtviewer
Text
Display
玄 丘


0
1
2
3
4