TR-82-017

中文字體產製後處理器

鄭國揚

陳振南

中華民國　七十一　年　六　月

前　言

中央研究院資訊科學所已初步完成中文字體自動產生系統 ACCFONT, ACCFONT 是一種根據字形組合之語法組成中文字的軟體輔助設計系統，其功能可分成三部份：(1)設計筆劃形狀，(2)根據字形組合關係的語法，筆劃被自動組成爲字根，(3)根據算子作用的語法，字根被自動組成爲中文字。ACCFONT 的特點是中文字的美觀性將完全依靠筆劃的設計方法，而和語法漸漸無關，換言之， ACCFONT 是屬於互作式的電腦圖形系統，根據互作式的經驗，組字的語法樹會漸漸地定型，附加在語法樹上的參數值將固定，每個中文字將對應一株標準的語法樹，不同字體形態的同一個中文字，其字形的差異僅是在筆劃形狀上的差異，而和其對應的語法樹無關。

ACCFONT 的筆劃設計是採用 B 軟樑函數論 [2] 的方法，任一筆劃是由一條或多於一條之三次 B 軟樑曲線所組成，所以在 ACCFONT 裏每個中文字的形狀，事實上是根據 B 軟樑公式計算得到。因爲從翻譯器(Translator)設計的觀點言，中文組字的語法樹是分析筆劃顯像窗在中文字顯像窗裏的位置，一旦每個筆劃的相對位置皆固定後，中文字本身就是由B 軟樑曲線群組成的函數，這種函數是以點向量的參數表示，因此，從事各種數學變換，如旋轉、轉移、放大縮小、透視投影的幾何變換，以及鏡射、倒影、變形等工作，是件容易的事情，由於它的中文字具有數學性， ACCFONT 系統的擴充將較容易符合中文版排的要求。

本文將依據 ACCFONT 的結果，設計一種可供中文排版用之後處理器 (Postprocessor)， 每個由 ACCFONT 製造之中文字形，經後處理器之編譯後，就產生各式各樣之輸出要求，諸如產生黑體字、浮體字、點矩陣碼、斜體字…等。從電腦圖形系統設計的觀點言 [3]， ACCFONT 所製造之中文字形爲圖形符號 (Symbols)，當這些圖形符號被加上定義其外表的參數後，就成爲圖形符號實體 (Symbol Instances)， 後處理器所處理的對象就是由圖形數據及非圖形數據所構成之圖形符號實體。

中文字形輸出之要求大致上可分爲：(1)字體的類別有宋體、仿宋體、明體、大小不同號碼之印刷體等，(2)字體的外表有黑體、浮體、斜體、點矩陣碼等，(3)字體的屬性有字體的尺寸、字的間隔、字體的品質等，(4)字體的傳送狀況有鍵盤、光筆、數位儀、檔案等輸入狀況，以及繪圖儀、印表機、圖形終端機、檔案等輸出狀況。這些要求可用參數來指定，這些參數加上圖形符號才構成顯像輸出時中文字的實體，所以後處理器的設計本身是一些演算法，輸入端是定義B軟樑曲線群的圖形數據以及供顯像用之非圖形數據，輸出端是中文實體的顯像。

本文將提出在各種不同字體輸出之要求下，如何規劃出有效的演算法，以及如何規劃成一個後處理器的系統，可和ACCFONT系統連接，俾能進一步地規劃中文排版之電腦系統。

ACCFONT筆劃之數學表示式

ACCFONT採用三次B軟樑曲線表示筆劃，有關B軟樑函數的理論以及三次B軟樑曲線的特性，請參閱參看資料[4,5]，以下我們僅提出均勻三次B軟樑曲線的計算公式如下：

$$B[P] = [t^3 \; t^2 \; t \; 1] \; \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_i \\ v_{i+1} \\ v_{i+2} \\ v_{i+3} \end{bmatrix} \qquad (1)$$

其中 t = FR(Ku)=Ku 的分數部份

i = INT(Ku)=Ku 的整數部份

K =總共的軟樑段 = 總共的控制點 -3

$u \in [0,1]$

B軟樑曲線是由有序的控制點群所決定，令 $P = \{v_1, v_2, \ldots, v_n\}$ 爲控制點群，$v_i \leq v_{i+1}$，$\leq$ 表示其順序，則P所決定的均勻三次B軟樑曲線是由(1)式所決定。

B 軟樑曲線在顯像時是以線段逼近，欲使得顯像輸出的曲線形狀保持平滑，則比較扭曲的部份需用較多的線段逼近，反之，則可用較少的線段逼近，這種方式的顯像輸出對於求軟樑曲線間的交點亦有助益，以下我們提出均勻三次 B 軟樑曲線之一種快速顯像法：

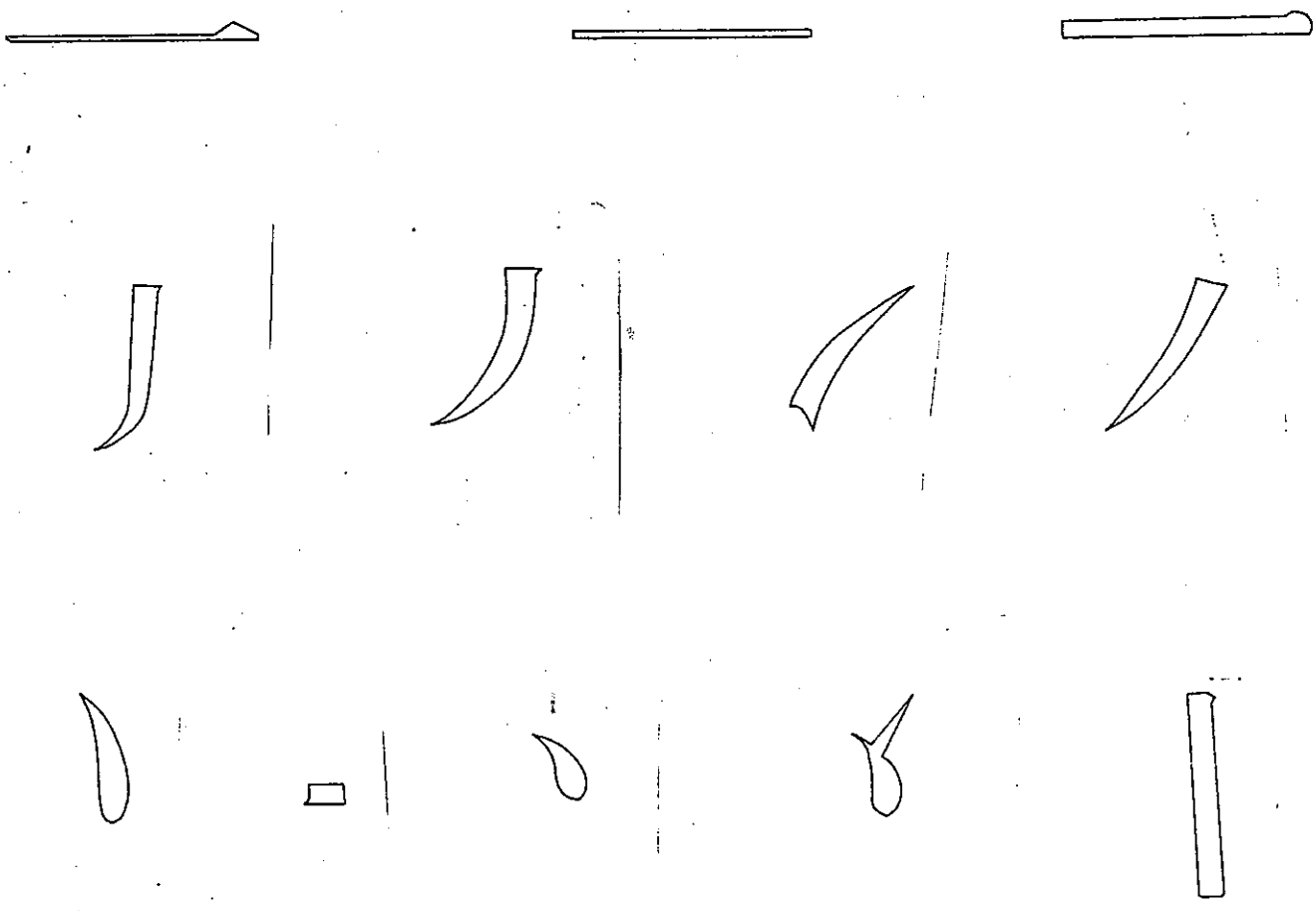
1. 計算第一條 B 軟樑段的首點並將之存入輸出暫存區。
2. 假如 n-3 條 B 軟樑段都已處理完畢則停止，否則繼續做以下的步驟。
3. 從 n-3 條 B 軟樑段中按順序挑選一條出來，並將其控制點群放在 $\{Q_0, Q_1, Q_2, Q_3\}$；
4. 計算由 $\{Q_0, Q_1, Q_2, Q_3\}$ 所決定的三次 B 軟樑段的首、尾點，P_0 和 P_1。
5. 求出 $\overline{Q_1Q_2}$ 與 $\overline{P_0P_1}$ 兩線段間的最大距離 d；
6. 如果 d 大於輸出儀器的解析度 (resolution) 時，用下式

$$Q_i^m = \frac{Q_i^{m-1} + Q_{i+1}^{m-1}}{2} \quad , \; i = 0,1,\ldots,4 \; , \quad m = 3,4$$

其中 $$Q_i^2 = \begin{cases} \dfrac{Q_i}{2} & , \; i \text{ 爲偶數} \quad i = 0,1,\ldots, 6 \\ \dfrac{\frac{Q_{i-1}}{2} + \frac{Q_{i+1}}{2}}{2} & , \; i \text{ 爲奇數} \end{cases}$$

則將 $\{Q_0, Q_1, Q_2, Q_3\}$ 細分成 5 個點群 $\{Q_i^4\}_{i=0}^4$ 使得前面 4 點再放入 $\{Q_0, Q_1, Q_2, Q_3\}$ 中，後面 4 點則存入疊堆中，再回到第 4 步驟繼續作。否則將尾點 P_1 存入輸出暫存區；再檢查疊堆中是否尚有資料；如果疊堆中是空的則回到第二步驟，否則從疊堆取出 4 個控制點，並將之存入 $\{Q_0, Q_1, Q_2, Q_3\}$ 中，再回到第 4 步驟。

圖一示出用均勻三次B 軟楔曲線爲公式所決定的筆劃，筆劃的顯像窗是取單位長的正方形，它們的一個筆劃經過ACCFONT之語法樹解析後，會在另一個定義中文字顯像窗中，佔據裏面的一個位置，使得該筆劃的顯像窗被放置在中文字顯像窗裏，因此它雖然經過解析改變了座標系，而且在新的座標系裏，控制點群的相對位置係依據語義動作的解析後經過調整，但是它的新控制點群用(1)式計算仍然是決定一條三次 B 軟楔曲線，而且是根據語義解析的動作產生其形狀，顯現在該筆劃的顯像窗裏，所以中文字的圖形符號定義是一群在符號顯像窗裏之三次 B 軟楔曲群的集合，換言之，每個中文字是由在一方塊裏的三次 B 軟楔曲線群所定義，以下我們將根據這項定義，提出各項字體外表的求法。



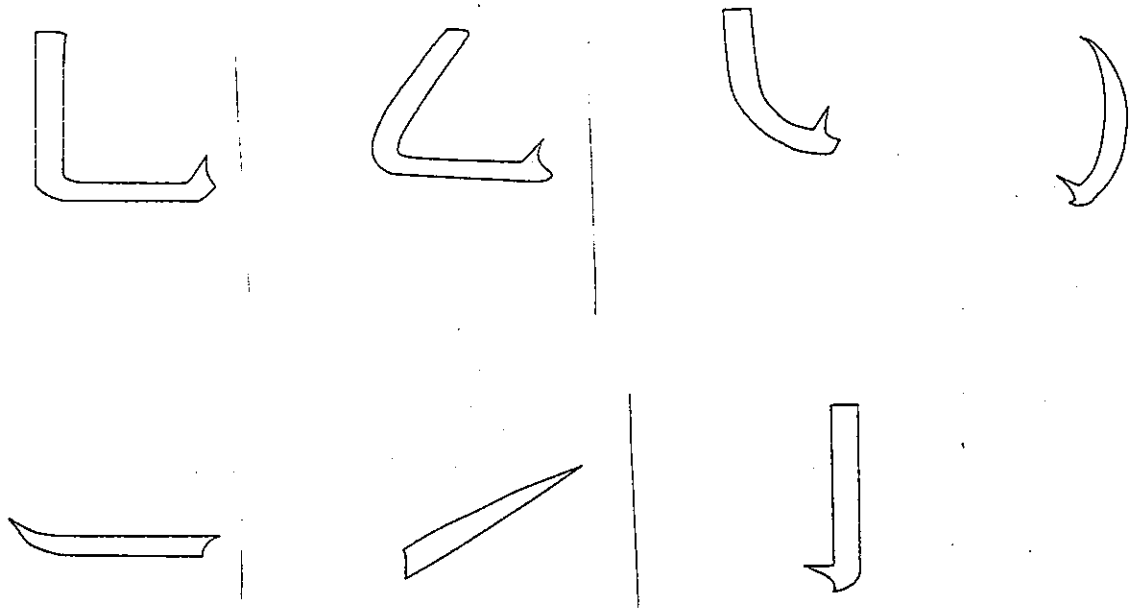


圖 一

黑 體 字 之 求 法

筆劃塗黑的方法有三類：(1)用直線段塗黑，(2)用曲線逐次內縮塗黑，(3)用點矩陣塗黑。決定用那一類的方法，端看對字體品質的要求以及輸出狀況而定，例如輸出至印表機就必須採用(3)類，輸出至繪圖機則可用(1)類或(2)類，至於選(1)類或(2)類就看應用時對字體品質的要求而定，通常，(2)類可達極高的品質，但是計算時間長，而(1)類的品質較差，惟計算時間快而且程式設計容易。

用直線段塗黑的方法簡單，其方法如下：(1)設定連接筆劃顯像的直線段數，令此段數的端點群爲 $\{x_i,y_i\}$，$i=0,1,2,\ldots,n$，其中 $(x_0,y_0)=(x_n,y_n)$，設定塗黑的筆數，則取垂直塗黑時，編排(Sort) x_i 成漸增的序列，將其元素依筆數的間隔各配成一組，將每組裏的最小 y 值和最大 y 值用線段連接。若取平行塗黑時，編排 y_i 成漸減的序列，序列裏的元素依筆數的間隔各配成一組，將每組裏的最小 x 值和最大 x 值用線段相連。圖二示出用直線塗黑中文字的結果，這種塗黑的方式所產生之黑體字

品質如何端看筆尖的粗細而定，筆尖愈粗，品質愈差，因爲從圖三可看出，走直線的筆尖常會產生超塗的現象，特別是筆劃比較扭曲的部份愈是嚴重。

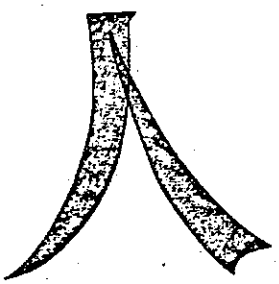


圖　二

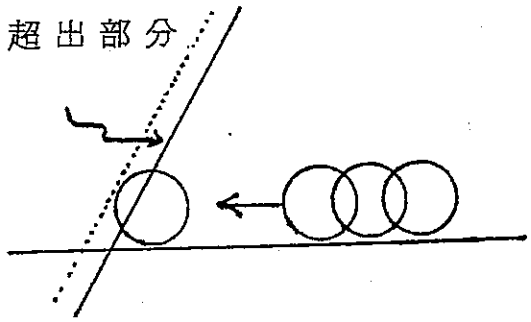


圖　三

用曲線逐次內縮塗黑可以避免超塗的現象，但是方法比較複雜，它需採用電腦動畫的方法，其方法如下：(1)設定筆劃逐次內縮的次數，(2)縮小筆劃控制點群所決定的控制多邊形至筆尖的直徑範圍內，並將縮小後的控制多邊形平移至原控制多邊形的重心，(3)呼叫動畫產生內縮次數的中間圖畫，有關電腦動畫的理論請閱參考資料［6］，圖四示出動畫塗黑的結果。

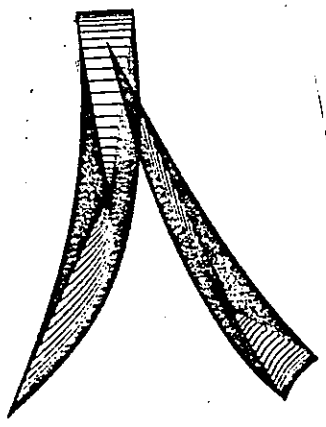


圖　四

用點矩陣塗黑的方法可敍述如下：(1)設定點矩陣的尺寸，令此尺寸爲$n \times m$，(2)令 (x_h, y_t), $h=1,\ldots,m$, $t=1,\ldots,n$ 爲點矩陣的元素，$x_h < x_{h+1}$，$y_t > y_{t+1}$ 令 (x_i, y_i), $i=0,1,\ldots,n$ 爲筆劃顯像直線段的端點群，則編排 x_i 成漸增的序列，將序列的元素依照 x_h 和 x_{h+1} 之間隔的左半或後半爲半徑的圓圈將它們分成 m 個區段，(3)令 $(x_h, y_t)= 1$（塗黑），若 x_i 落在 k 區段而且 $|y_i - y_t| < r$，$r = \frac{1}{2}|y_t - y_{t+1}|$，否則 $(x_h, y_t)= 0$（不塗黑），圖五示出點矩陣塗黑的結果。

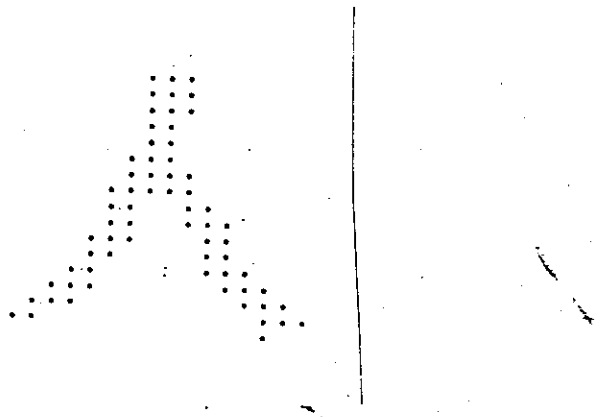


圖　五

點矩陣塗黑會造成中文字的外表變形，特別是點矩陣的尺寸愈小時，外觀愈難控制，所以，若要求更高的字體品質時，上述的方法就不敷應用的要求，吾人必須用更複雜的演算法處理，關於此種演算法的開發將列爲未來研究的項目之一。

浮　體　字　之　求　法

從 ACCFONT 輸出之圖形符號定義的中文字，筆劃是重疊在一起的，其情形如圖六所示，產生浮體字的要求是去除重疊的部份，使其結果如圖七的樣子，從互作電腦圖形顯像的觀點言，這種問題爲隱藏線演算法 [7] 所討論的範疇，因爲若一筆劃先當作另一筆劃的背景圖，則另一筆劃被遮隱的部份亦可用相同的處理去除。

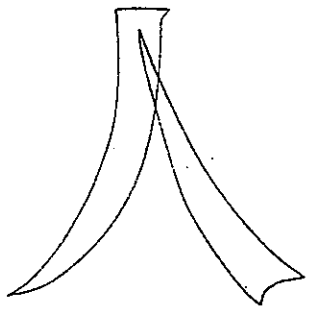


圖 六

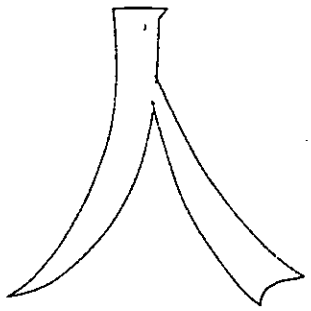


圖 七

讓我們首先探討用一般隱藏線演算法求浮體字的可行性爲何，令 (x_i, y_i), i =0,1,..,n, $(x_0, y_0)=(x_n, y_n)$，爲一筆劃顯像之直線段的端點群，$(\bar{x}_j, \bar{y}_j)$, j=0,1,...,m, $(\bar{x}_0, \bar{y}_0)=(\bar{x}_n, \bar{y}_m)$，爲另一筆劃顯像之端點群，則用一般隱藏線演算法，需求 $n \times m$ 條直線的交點，其計算量是相當可觀的，雖然可以先用極大極小測試，將部份不可能相交的直線先行判斷，再對可能相交的直線段做處理，但是即使這樣去做，求交點的計算工作仍然是相當花時間的，特別是求二條可能相交而實際上不相交的直線時，所以用一般隱線演算法來求中文的浮體字是不切實際的方法，必需另想辦法以減少大量無謂的計算。

事實上，有幾個辦法是可以達到減少大量無謂的計算，在此我們僅指出二種可行的辦法，第一種辦法稱爲直接法，第二種辦法稱爲遞迴內含測試法。直接法係利用 ACCFONT 語法樹解析的一些參數值決定那些筆劃顯像直線段最可能相交，因此很快地將交點求得，遞迴內含測試法係利用顯像窗重疊部份才有可能相交的理由，將顯像窗重疊處遞迴求出，因此亦可很快地將交點求得，以下我們將這二種辦法詳述之。

從語法樹的解析，我們可獲知二個筆劃間相交的位置，令二個筆劃接合點爲 (x_h, y_{t_i})而且第二個筆劃之曲線起點被包含在第一個筆劃之內，則先以 (x_h, y_t) 爲原心，它和第二個筆劃之曲線起點的連線爲半徑畫一圓圈，令 R 爲此圓圈的範圍，則

二個筆劃分別之顯像直線段端點是否落在 R 裏可很快獲知，只有其中一個或二個端點落在 R 內的直線段最有可能相交，因此，我們只要對這些線段求交點即可，一旦求得交點，則第一筆劃的 B 軟樑曲線從起點畫起碰到奇數次的交點就提筆（隱藏線）碰到偶數次的交點就下筆（畫線），而第二筆劃的 B 軟樑曲線從起點畫起時就先提筆碰到奇數次的交點就下筆，而碰到偶數次的交點就提筆。這種方法的缺點是筆劃間的交接點必需事先規劃周詳，否則這種方法有時會失敗。

遞迴內含測試法一定可以找到筆劃相交之顯像直線段的範圍，讓我們用圖八說明這種方法的構想，圖八 (a) 示出二個筆劃分別存在的顯像窗，它們顯像窗重疊很容易可以算出，而且顯然地可以看出，顯像直線段僅會在重疊處相交，圖八 (b) 示出再一次遞迴求出重疊的顯像窗，這是將二個筆劃可能相交的部份分別訂出顯像窗，再求它們的重疊部份而獲得的，圖八 (c) 示出再遞迴一次求出重疊顯像窗的情形，可以看出，此重疊顯像窗再也不能遞迴細分了，因此，最後這個最小的重疊顯像窗所包含的顯像直線段才有可能相交，落在此顯像窗外的顯像直線段絕不可能彼此相交。

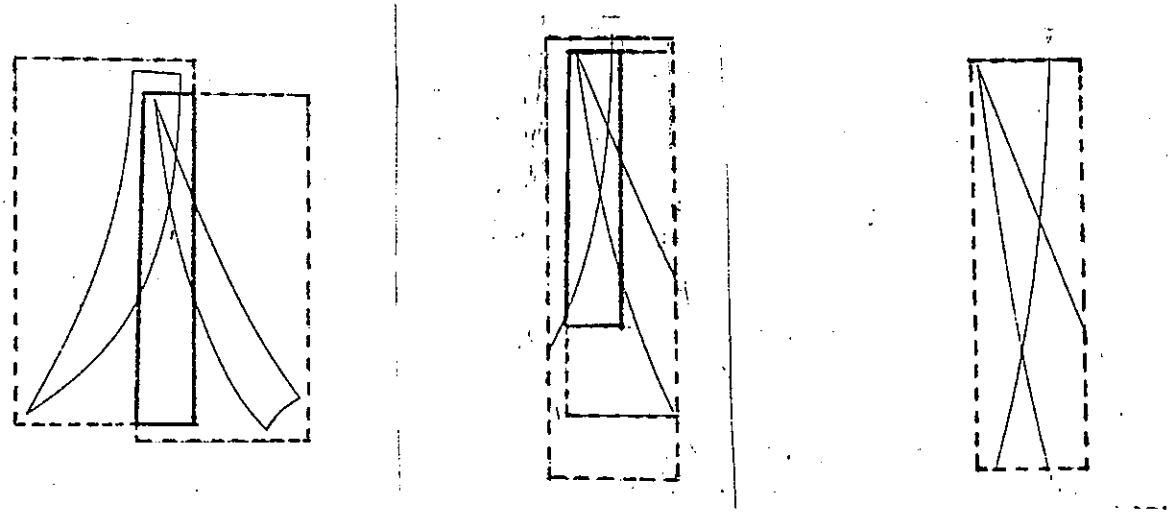


圖　八

從以上的說明可知，遞迴內含測試法最主要的工作是作顯像窗剪除(clipping)　亦即判別那些顯像直線可能落內顯像窗內，剪除的動作可以用邏輯判別的方法，例如可採用九瓦法[8]其法可用圖九說明，令九瓦中央的那塊瓦爲重疊的顯像窗，其他八瓦的建立是依據此瓦的上下左右部份而定，例如一點若落在顯像窗之上方及左邊，則此點是落在左上角的那塊瓦，以此類推，則每個顯像直線段的端點皆可知它們落於那塊瓦，同時我們假設任一條顯像直線段不會跨越三個瓦（因爲 ACCFONT 的結果不會產生這種現象）。

	x_0x_1	$\bar{x}_0\bar{x}_1$	$\bar{x}_0x_1$
$\bar{y}_0y_1$	1101	0001	0101
$\bar{y}_0\bar{y}_1$	1100	0000	0100
y_0y_1	1111	0011	0111

圖　九

利用邏輯線路的觀念，令重疊顯像窗的邏輯值爲 $\bar{x}_0\bar{x}_1\bar{y}_0\bar{y}_1$，一點 (x_i, y_i) 落在其內時，則此點的邏輯值爲 0000 同理，該點若落在左上角那塊瓦，則其邏輯值爲 1101 其他依此類推。根據這樣的安排，我們可知某點一定落在重疊顯像窗內，若其

$$x_0 \vee x_1 \vee y_0 \vee y_1 = 0$$

令 P_i 和 P_j 分別是一條顯像直線段的端點，它們的邏輯值分別是 $x_0^i\ x_1^i\ y_0^i\ y_1^1$ 和 $x_0^j\ x_1^j\ y_0^j\ y_1^j$，令 $C^i = x_0^i \vee x_1^i \vee y_0^i \vee y_1^i$ 以及 $C^j = x_0^j \vee x_1^j \vee y_0^j \vee y_1^j$，則做下面的測試：

計算 $P_1=\bar{C}^i \wedge \bar{C}^j$ 和 $P_2= C^i \wedge C^j$,若 $P_1=1$ 則此線段完全落入重疊顯像窗，若 $P_2=1$ 則此線段一定落在顯像窗之外，若 $\bar{P}_1 \wedge \bar{P}_2=1$，則此線段和顯像窗有交點，因爲我們已知 P_1 和 P_2 的落點在何處，所以交點是在顯像窗的那條邊線 ($x = x_{min}$, $x = x_{max}$, $y = y_{min}$, $y = y_{max}$) 上很容易判斷，所以交點很容易算出，而且此線段就被分成二段，一段落在窗內，一段在其外，保留其內者，去除其外者。

根據上面的測試步驟，每條顯像直線段的落點狀況可知，於是就可完成顯像窗剪除的工作。

特殊字體之求法

正楷的中文字體是取正方形的顯像窗擺放，但是斜體字則是取左右二邊傾斜的平行四邊形爲顯像窗，同理，其他特殊字體亦可取左右爲相同形狀之曲線而上下是平行的四邊形狀爲顯像窗，讓我們稱經由窗函數轉換之中文字形爲特殊字體，則其求法可敍述如下：

令一新顯像窗左右二邊爲一相同形狀的曲線，左邊的曲線爲一條均勻三次 B 軟樑曲線，它由控制點群 $\{P_i\}$, $i = -1,0,1,\ldots, n,n+1$, $P_i= (\bar{x}_i,\bar{y}_i)$ 所決定，若 $v_j =(x_j, y_j)$ 爲在原顯像窗裏的一個控制點，此點經窗函數 "ω：原顯像窗 → 新顯像窗" 轉換，則其在新顯像窗的位置變成爲

$$(x_j + \alpha_j , y_j)$$

其中 α_j 的值可由下面的程序獲得：

$$(1) \quad y_j = [t^3\ t^2\ t\ 1]\ \frac{1}{6}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix}\begin{pmatrix} \bar{y}_i \\ \bar{y}_{i+1} \\ \bar{y}_{i+2} \\ \bar{y}_{i+3} \end{pmatrix}$$

$t = FR(nu)$, $i = INT(nu)$, $u \in [0,1]$

決定 t 和 i 的值，再將此 t 和 i 的值代入下式

$$(2)\quad \alpha_j = [t^3\ t^2\ t\ 1]\ \frac{1}{6}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix}\begin{pmatrix} \bar{x}_i \\ \bar{x}_{i+1} \\ \bar{x}_{i+2} \\ \bar{x}_{i+3} \end{pmatrix}$$

假設新顯像窗和原顯像窗的 (0,0) 點重合，而且座標採用原座標系，則求斜體字時，新顯像窗的左右二邊各是一條斜線，設其和x 軸的夾角爲 θ，則上式可簡化得 $\alpha_j = y_i/\tan\theta$ 因此，我們可用 $(x_i + y_i/\tan\theta,\ y_i)$ 爲控制點 v_i 在新顯像窗的位置，圖十示出產生斜體字的過程。

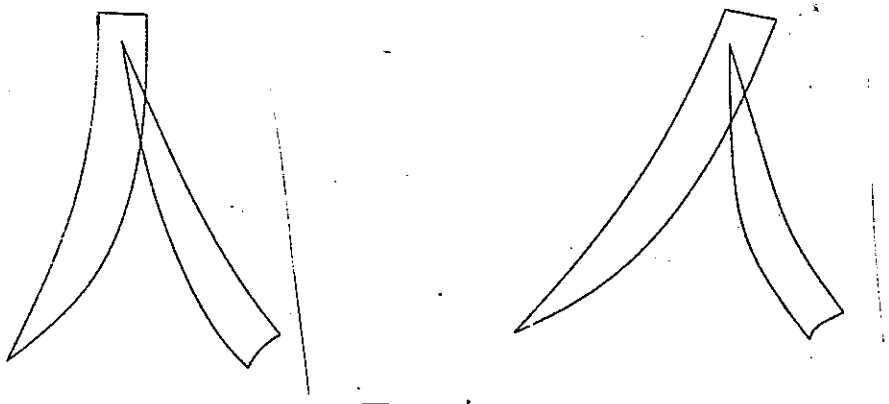


圖 十

產生特殊字體的窗函數並不限於上述對稱的顯像窗，事實上，任何的窗函數皆可用，只要使用人能夠掌握中文字形和窗函數間的關係，譬如說窗函數將原來是正方形的顯像窗轉成四邊是曲線形狀的顯像窗，曲線形狀左右一樣且上下一樣時，則我們可以分成二次處理：(1)處理 x 軸的轉換時，將新的顯像窗看成上下是直線平行的，原控制點經轉換後，再到下一步驟去處理，(2)處理 y 軸的轉換時，將新的顯像窗看成左右是直線平行的，將步驟(1)所得到的控制點經此窗函數的轉換後，即得到原控制點經窗函數映像 (mapping) 至四邊皆曲線的顯像窗。

從以上的敍述知，求特殊字體的數學運算是一些轉換式，其中看起來似乎是以解從已知y_j值求出參數 t 和 i 最困難，但是我們採用均勻三次 B 軟樑曲線當做顯像窗的邊線，從其特性可知曲線上之各個軟樑段的範圍，由此，可以立即判別出 i 的值，因此參數 t 的值可從解三次方程式的根獲得，所以整個求特殊字體的轉換過程並不困難。

點矩陣加碼法之探討

中文字之點矩陣加碼和用印點塗黑中文字之問題是一樣的，只是點矩陣加碼是輸出十六進位的數值，而印點塗黑是將點矩陣直接輸出，它們皆面臨一個問題：點矩陣中文輸出字體的品質怎樣？特別是點矩陣$m \times n$的尺寸亦會影響其品質，怎樣才能令不同尺寸的點矩陣皆可得同等級的品質？這些問題曾一再地被研究，在這方面有很多論文 [9,10] 陸續地發表。我們的問題是：中文圖形符號的定義已知是用均勻三次 B 軟樑曲線群表示，其字形是絕對高的品質，用點矩陣處理於是破壞了其品質，有沒有辦法利用其字形定義的數學性，使字體品質被破壞的程度減至最低？

點矩陣字體的品質大致上可用下列特性描述：(1)整個字的對稱性（這種對稱性包括垂直的、水平的、對角的），(2)每個筆劃的對稱性，(3)不連接筆劃間的分離度，(4)相對的尺寸（字的高度與寬度、筆劃的寬度、筆劃之間的間隔），(5)邊線的平滑性，(6)轉角的方正性（包括內部與外部），(7)每個字根底線的正確性，…等。這些特性如何從均勻三次軟樑曲線的數學運算中以及點矩陣的顯像處理中獲知，目前尚未深入研究，不過這項探討將是非常重要，非常值得去做，本文不擬再加以討論，擬留待未來研究。

結　　論

ACCFONT 中文字體自動產生系統裏的圖形表示採用了均勻三次 B 軟樑曲線，經由語法組字的解析後，產生了定義中文字的圖形符號，這些圖形符號僅是筆劃形狀重疊在一起的圖形，因此我們說 ACCFONT 的輸出結果是定義中文字的圖形符號，欲將中文字依照各種需求的規格顯現，必需有一個後處理器的軟體程式來處理，從本文的敍述中可看出，這種後處理器主要是一些圖形處理的演算法，爲了配合 ACCFONT 的圖形處理方式，這些圖形處理的演算法仍然是以均勻三次 B 軟樑曲線的公式爲主。

ACCFONT 產生之中文圖形符號本身是相當美的，如果這種美觀性在後處理器的處理過程中不被改變時，例如黑體字、浮體字、斜體字等筆劃的連續性一直保持不變，那麼後處理器的演算是根據均勻三次 B 軟樑曲線表示式做數學轉換的處理即可。反之，這種美觀性在後處理器的處理過程中需遭破壞時，例如點矩陣塗黑或加碼等筆劃的連續性被破壞了，那麼爲了保持字體輸出的品質，後處理器的設計就必需用更複雜的處理方法，這種處理方法有時需涉及顯像處理的經驗法則 (Heuristic Algorithms)，本文對這種處理方法未加以深入探討，乃擬根據參考資料 [11] 的一些技巧開發可產生高品質的點矩陣字體。

中文字體輸出之後處理器的軟體設計，應與互作式電腦圖形標準性的處理一起考慮，將來較易和其他電腦圖形系統連接，可以變成中文與圖形之文件處理系統，由於中央研究院資訊科學所目前正在設計一套互作式電腦圖形系統，它是採用 ACM SIGGRAPH 圖形委員會所製訂之 CORE 標準系統，所以我們在中文字體輸出後處理器的設計上，亦完全採用 CORE 的圖形定義、圖形參數、以及圖形副程式的製訂方法。

中文字體輔助設計系統包括了中文字體自動產生系統(ACCFONT)中文字體輸出後處理器，以及交談式的前處理器 (preprocessor)，有了這樣的一套軟體輔助設計系統，對於許多中文電腦輸出的

設計有很大的助益，因爲只要鍵入字體的類別、字體的外表、字體的屬性，以及字體輸出的狀況，就會獲得中文輸出的結果，同時，它對於未來發展中文文件處理系統（包括中文排版系統）亦有很大的助益，因爲這些系統要求同時產生多種字體的類別，多種字體的尺寸，以及美麗的字形。

由於價格上的考慮，中文輸出儀以印表機爲主，因此中文輸出以點矩陣最爲普遍，怎樣從均勻三次B 軟樑曲線產生高品質的點矩陣中文字體，將是未來研究設計中文字體輸出後處理器最主要的課題。

參 考 資 料

1.鄭國揚、陳克健，Account - An Automatic Chinese Character Font generating system，Technique Report 中央研究院 1982.

2. Gordon, W. J. and R. F. Riesenfeld, "B-spline curves and surfaces", Computer Aided Geometric Design, edited by R. E. Barnhill et al., Academic Press, pp.95-126, 1974.

3. Giloi, W. K., Interactive Computer Graphics, Prentice-Hall, Inc., Englewood Cliffs, N. J.

4. Coons, S. A., Surface patches and B-spline curves, in Computer Aided Geometric Design, edited by R. E. Barnhill et al., Academic Press, pp.1-16, 1974.

5. Jeffrey, M. L. and R. F. Riesenfeld, A theoretical development for the computer generation and display of piecewise polynomial surfaces, IEEE Tran. on PAMI, No-1, January 1980.

6.鄭國揚，基於 B 軟樑理論之電腦卡通製作系統，中國工程學刊，第四卷，第一期（民國七十年）。

7. Sutherland, I. E. and R. F. Sproull, A clipping divider, Pro. AFIPS EJCC 1968, 765.

8. Sutherland, I. E., R. F. Sproull and R. A. Schumacker, A characterization of ten hidden-surface algorithms, computing surveys 8,1, Mar. 1974, pp.1-55.

9. Yaohan Chu, Chinese character dot-matrix printers, proc. of ICS, 1977, pp.186-208.

10. R. G. Casey, Use of pattern processing techniques to scale digital print fonts, IBM Research Laboratory, 1980.

11. R. D. Bergoron, Graphics programming using the core system, Computing surveys, Vol.10, No.4, December 1978, pp.389-443.