Journal of Science and Engineering Technology, Vol. 1, No. 2, pp. 13-22 (2005)

文件影像之手寫中文字擷取技術

曾逸鴻
大葉大學資訊管理學系
彰化縣大村鄉山腳路 112 號

摘　要

光學手寫字的資訊不像印刷字般的穩定性，因此對於手寫稿的文字擷取必須有別於印刷文件的擷取方法。傳統的文字擷取方法，不外乎是利用相連元件偵測法和投影輪廓分析法，來找出可能的文字區塊位置。由於手寫文件的傾斜校正不易，前者的方法通常比後者有較佳的文字抽取結果。在本研究當中，我們先利用堆疊方式偵測出所有相連元件，而後提出三階段的元件合併過程，利用元件間彼此的重疊性、元件間的間隔大小、與元件兩側鄰居間隔差異性，來決定兩元件是否合併。最後再利用字元區塊投映與鄰近元件位置差異檢測方式，決定出所有字元的讀序。在實驗過程，我們請 15 人寫了 57 張手寫稿，合計 1148 字元。以掃描器數位化成 tif 格式影像後，利用本系統作字元抽取，可得到 98.43% 的正確抽取率。

關鍵詞：相連元件，投影輪廓，重疊性，間隔差異性，區塊投映，讀序

Handwritten Chinese Character Extraction from Document Images

YI-HONG TSENG
Department of Information Management, Da-Yeh University
112 Shan-Jiau Rd. Da-Tsuen, Changhua, Taiwan

ABSTRACT

Information in handwritten characters is unstable when compared to printed characters in document images. The extraction approach for handwritten characters must be different from that for printed characters. Traditionally, connected component detection and projection profile analysis have been two common approaches for determining character positions in document images. Since the skew-detection result from handwritten document images is unsatisfactory, the former extraction approach can obtain better results than the latter. In this study, connected components are detected by using a stack-based approach, and a three-stage process is proposed to merge components belonging to the same character. Overlapping of components, a gap between components, and an opposite-gap ratio of components are used to determine the possibility that two components may be merged. Finally, character-block mapping and position difference among neighboring components are used to determine the reading order for the extracted characters. In this experiment, 1148 characters in 57 manuscript documents written by 15 persons are used to test the proposed technique.

The extraction rate is 98.43%.

Key Words**:** connected component, projection profile, overlapping, opposite-gap ratio, block mapping, reading order

一、緒論

近年來電腦的技術日新月異，電子書、電子報等電子文件帶給出版業極大的衝擊，也讓人們對未來無紙張世界充滿了期待。然而，對於現存的大量紙張式文件，如何將其數位化，以方便保存與快速流通，則是愈顯重要的課題。將整張文件掃描經過壓縮存成影像檔是個解決法，但仍嫌太占空間，且不易修改。若將文件中影像部分挖出壓縮，文字部分以字碼方式儲存，則不但節省大量空間，且新增、刪除或修改文字內容均極爲容易。文件分析與文字辨識的研究，也應運而生。在一個文件自動處理系統（automatic document processing system）中，大量的現存文件（如報章、雜誌、公文、表格等）可先利用影像擷取設備（如數位相機、掃描器等），將紙張式文件數位化成爲文件影像，爲求日後的文件檢索需求，進而使用光學文字辨識（optical character recognition, OCR）技術，來將文件影像中的文字內容辨識後，存成文字碼。日後需要檢索時，即可讓使用者輸入搜尋字串，來與文件內容進行字串比對。

光學文字辨識（OCR）的研究幾年來受到不少關注[5,9,17,18]。文件影像需先經過文件分析（document analysis）模組 [10,19,22] 來分離文件中的文字、影像、表格等不同類型區塊。而一些額外處理，如雜訊去除 [8]、傾斜校正[1,7,12,21]、表格結構抽取 [3-4,11] 等，也經常在處理一些常用格式之文件時會用到。經過文件分析後，需將區塊中文字一一擷取，以送入辨識核心。爲加強辨識核心應用與提升辨識效果，在辨識之前，可能還需判斷該文字影像是手寫字或印刷字 [6]，如果是印刷字，還需判斷是何印刷字體[14,25]。如此一來，可將手寫字以手寫辨識核心辨識，而印刷字根據不同字體，採用不同的印刷字辨識核心。

文字切割的效果與辨識效果有很直接的影響，如果將多字合成一字或者將一字切成多字，都會嚴重地影響文字辨識的準確性。Lu [15] 評論一些印刷文件切字的相關論文，探討了包括一致或成比例字體的字元切割、破碎字的處理與相連字的切割等，利用文稿特性及辨識結果的技巧也有所探討。Lu and Shridhar [16] 也評論了一些手寫字的切割方法，主要在探討工整手寫字（handprinted word）、手寫數字（handwritten numeral）、與草寫文字（cursive word）等的切割。Casey and Lecolinet [2] 將切字的方法大概分成三大類：解剖方法（dissection methods）、整體方法（holistic methods）和辨識基礎方法（recognition-based methods）。第一類的解剖法先將文字行影像切成數個有意義的元件（components），這些元件需滿足事先定義好的文字特性（如字寬高，字距等）；第二類的整體法則將整個詞（word）直接做辨識，不經過切字，此方法較適合以詞爲句子基本單位的語系（如英文），中文則以字（character）爲句子組合單位；第三類以辨識爲基礎的方法先找出一些可能的切割點，再利用辨識的結果來檢驗這些切割點的正確性。一般來說，第二類方法並不適於用在中文字的切割，因爲中文字的組合太多樣化了。第三種方法通常會得到較好的結果，但是需有良好的辨識核心作爲輔助，較適合印刷字的切割。雖然手寫字的切割也可用第三類方法，假設相鄰字元碰觸（touch）不嚴重的情況下，且手寫辨識核心尚未有穩定且令人滿意的結果前，實做可用的手寫文字抽取技術仍以第一類方法爲優先考量。

要將文字區塊中擷取出字元位置，可能的字元位置決定有兩個主要的方法：投影輪廓分析（projection profile analysis）和相連元件偵測（connected component detection）。前者將所有黑點往 X 軸及 Y 軸做投影，將累積黑點數目小於某個門檻值的位置視爲可能的切割點。後者將八方向相連（8-connected）的黑像素點（black pixels）予以連接，成爲一個相連元件（connected component），這些元件的邊界即可視爲切割位置。然而，利用這些方法只能找出線性（linear）的切割線，對於一些傾斜字或字距較近的手寫字，往往無法利用直線來決定字元位置，因此有些應用需找出非線性（non-linear）的切割路徑（segmentation path）[26]。Wang and Jean [27] 對碰觸字（touching characters）利用最短路徑偵測法（shortest path detection），找出一個非線性的切割路徑。對於水平的文字行（text line），只有往下、往左下及往右下三個方向被考慮。路徑通過黑點比通過白點有較大的代價（cost），而且有額外的處罰（penalty）加諸在對角的移動（即往左下或右下）。Lee *et al.* [13] 在灰階圖上利用分析投影量

（projection profile）及地形特徵（topographic features），先找出切割路徑可能出現的範圍，然後利用一個多階段圖形搜尋演算法（multi-stage graph searching algorithm），在每個可能範圍內找到一條非線性的切割路徑（non-linear segmentation path）。當然，以上述所找出的非線性切割路徑並不一定是正確的，還是需利用一些文字特性或辨識結果來幫助驗證其正確性。另外，由於中文字可說由八種不同方向之筆畫所組成，對於一些碰觸的字元或者同字元分離的部首，也可利用字元筆畫的位置與方向性，來決定同字部首的合併與碰觸字元的分離 [23]。

文字的連筆狀況，一直是文字抽取研究中，相對的困難課題。可利用投影量分析、兩端極點距離、通過最少黑點數量等方式，找到一些可能的切割位置，再利用文字辨識核心，來作切割位置的決定。然而，連筆的情況可能是兩字相連或多字相連，利用長寬比的判斷，或許可以預估相連的字數。然而，對於以部首組成的中文字而言，可能前一字的右部首與後一字的左部首相連，但同一字的部首卻是分離的。例如手寫的「日王月半」，如果考量可能的連筆狀況，即使是利用辨識效果來協助文字切割，還是無法百分百確定是「日王月半」、「旺月半」、「日王胖」、「旺胖」、「日玥半」何種切割方式。其實各類的文字切割問題都有特定的複雜解法，因爲手寫習慣或手寫字的變異特性，如何不使用辨識核心的情況下，先將絕大部分手寫字的組成部首合併正確，是我們實務研究的重點。後續視需要再利用複雜的切割位置確定方法，將連筆的情況解決。

不同的文字擷取方法，針對不同的應用，各有其實用性。但是相對的，文字擷取的效果與速度效率，也有所不同。本研究的目的，是將在白紙上所寫的手寫稿，經過掃描成文件影像後，將個別字元抽取，經過分行後，利用字元位置決定讀序（reading order），並以此順序將擷取出的手寫字影像，送入手寫辨識核心 [24]。由於已經假設字元的碰觸情形不嚴重，而且手寫稿的字元通常不會很對齊，因此我們採用相連元件偵測的方法找出所有相連元件，並利用統計出的文字特性（如平均字元寬高、平均字元間距等），訂定幾個合併條件，將同一字的幾個分開元件予以合併。再來利用區塊對映（block mapping）方式決定哪些字元屬於同一行，並判斷字元間是否插有空白間隔。經過文字擷取模組的處理後，手寫文稿的文字內容可分成三類：一般字元、空白間隔、換行記號，再將擷取出的一般字元送入手寫字辨識核心。

本論文後續的第二節將提出相連元件的擷取方式，第三節會介紹利用三階段式相連元件合併方式，完成手寫文字擷取，第四節會說明手寫稿的讀序決定方式，第五節將解釋實驗結果並提出我們的分析，最後第六節將提出此研究的結論。

二、相連元件偵測

對一般的手寫稿文件通常並不花俏，而是以淺底深字方式，以深色筆在淺色紙張中書寫，手寫稿文件經過掃描，若爲彩色或灰階影像，需先經過二值化 [20]，轉成黑白影像後。我們將在黑白影像中，進行以堆疊爲基礎的相連元件偵測方法（stack-based connected component detection）。

首先，我們在影像中找到任一黑點（如圖 1 之中心點 (x, y)），檢查該點周圍的八個鄰居（即圖 1 的 $(x-1, y-1)$, $(x, y-1)$, $(x+1, y-1)$, $(x+1, y)$, $(x+1, y+1)$, $(x, y+1)$, $(x-1, y+1)$ 與 $(x-1, y)$），若也是黑點，則將該鄰居位置放入（push）堆疊中。然後從堆疊中提出（pop）一個黑點，再將此黑點的鄰居黑點位置放入堆疊中。重複上述動作，直到堆疊中無任何黑點。所有放入堆疊中的黑點，都是跟堆疊中其他黑點有八方向的相連性，我們就可以將曾放入堆疊中的黑點，記錄其出現範圍，即可找到一個相連元件。若此黑白文件影像中，還有尚未看過的黑點，則重複上述的堆疊法，我們即可將文件中所有相連元件找到（如圖 2 所示）。

由於文件原始品質或影像二值化的變異，有些黑白文件影像抽出的相連元件包括了一些小雜訊，或者兩小元件重疊性高或距離近，其實應屬同一文字。因此，對於利用堆疊法偵測出的相連元件，需做進階的過濾與合併動作，才能將手寫字元完整擷取出來。首先，將過小的相連元件當作是雜訊去除，然後將重疊性高的元件合併。再來統計手寫稿文件中的平均元件寬度與高度，還有與鄰近元件的平均水平和垂直間隔，利用這些特性，將同一文字行中距離夠近且合併後滿足限制的元件予以合併。則會得到較能接受的字元擷取結果。

(x-1, y-1)	(x, y-1)	(x+1, y-1)
(x-1, y)	(x, y)	(x+1, y)
(x-1, y+1)	(x, y+1)	(x+1, y+1)

圖 1. 抽取相連元件用之 3x3 區域

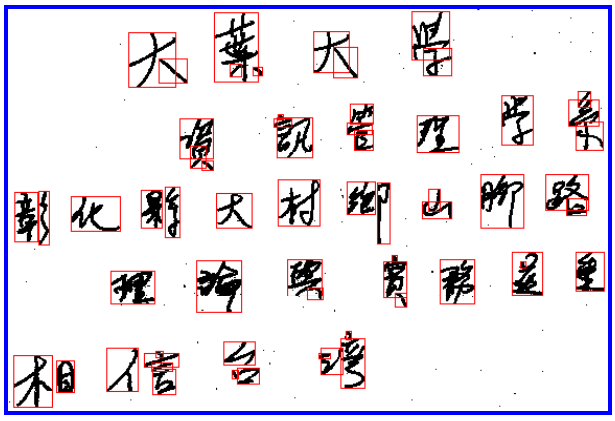

圖 2. 相連元件偵測結果

三、三階段式之元件合併

由於中文字通常由一些並不相連的元件（或稱部首）組成，部首（radical）間可能因有間隔（gap）而導致被當成獨立的元件，因此需要利用一些條件設法將其合併。在此元件合併模組中，我們以三階段（stage）的合併方式，將分開的部首合併（merge）成同一個字元。

（一）第一階段元件合併

從圖 2 所顯示的相連元件偵測結果，我們可觀察到有些元件必定要合併的，例如某元件 A 完全包含在元件 B 中，或者元件 A 與元件 B 有相當大的重疊比例。因此，在我們的第一階段元件合併，重疊性（overlapping）是最主要的合併條件考量。

對於每個元件 A，其邊界爲 $Left_A$、$Right_A$、Top_A 和 $Bottom_A$，寬度與高度爲 $Width_A$ 和 $Height_A$。考量其他元件與元件 A 的重疊性，假設有一元件 B（邊界爲 $Left_B$、$Right_B$、Top_B 和 $Bottom_B$，寬高各爲 $Width_B$ 與 $Height_B$），元件 A 與元件 B 的重疊區域 OVP，其寬度與高度計算方式如下：

$$Width_{OVP} = Width_A + Width_B - \left(\max\left(Right_A, Right_B\right) - \min\left(Left_A, Left_B\right)\right) \quad (1)$$

$$Height_{OVP} = Height_A + Height_B - \left(\max\left(Bottom_A, Bottom_B\right) - \min\left(Top_A, Top_B\right)\right) \quad (2)$$

AB 兩元件的重疊性 $Value_{ovp}$ 計算如下：

$$Value_{OVP} = \max\left(0, \frac{Width_{OVP} \times Height_{OVP}}{\min\left(\left(Width_A \times Height_A\right), \left(Width_B \times Height_B\right)\right)}\right) \quad (3)$$

如圖 3(a) 所示，元件 A 與元件 B 的重疊性，即是將重疊區域的面積除以兩元件的面積小者。若兩元件完全重疊（如圖 3(b)），則重疊性爲 $Value_{OVP}=1$；若兩元件完全分離（如圖 3(c)），則重疊區域 OVP 的寬或高爲負值，重疊性爲 $Value_{OVP}=0$。當重疊性高於否門檻值（$Value_{OVP}>Thr_{OVP}$），即代表元件 A 與元件 B 應該合併。

這種以重疊性爲基礎的合併方式具有遞移性，也就是如果 A 與 B 合併，且 B 與 C 合併，則 A 與 C 也應合併。因此我們建立一個合併陣列 $MrgAry$，長度爲全部的元件數目 n。一開始 $MrgAry[i]=i$，$i=1,2,...n$，兩兩元件利用重疊性偵測是否合併，若元件 A 與 B 需合併（假設 $A<B$），則將兩元件的 $MrgAry$ 值設成一致：$MrgAry[B]=MrgAry[A]$。所有元件都檢視完畢，即可將 $MrgAry$ 內所有值相同的元件予以合併。另外，合併後的元件尺寸變大後，可能有機會將一些本來不重疊的元件再併進來，因此我們需作重複合併（recursive merging）的動作，直到沒有元件可再被合併爲止（即 $MrgAry$ 內找不到值相同的兩元件），會得到第一階段的元件合併結果（如圖 4 所示）。

（二）第二階段元件合併

在第一階段元件合併中，對於重疊性夠高的元件，可以合併爲同一元件，並以重複偵測重疊性方式，遞迴地將重疊的元件合併。即使已經將重疊性夠高的元件予以合併，仍然還有一些文字本來就是由多個分離的部首（radical）組成，只利用重疊性無法將這些部首元件合併成同一字。因此，我們在第二階段的元件合併過程中，將考慮元件大小與元件間隔等特性，將距離夠近，且合併後滿足條件的元件予以合併。

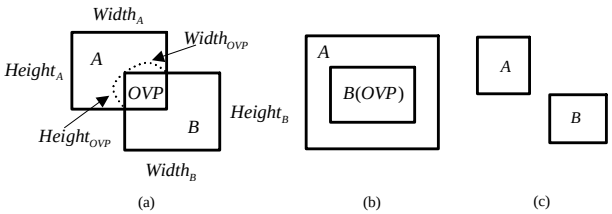

圖 3. (a) 兩元件含有部分重疊區域；(b) 兩元件完全重疊；(c) 兩元件完全分離

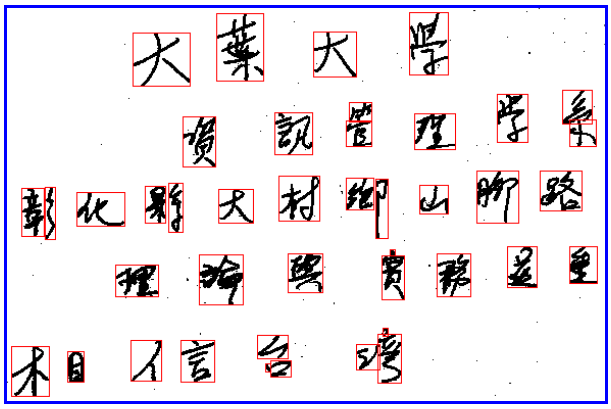

圖 4. 第一階段的元件合併結果

1. 字元特性分析

我們假設同一中文手寫稿中，應該是同一人在同一時間所寫，內容字元的差異性應該不大。因此，在第二階段元件合併處理之前，要先將一些字元特性予以統計並分析。在本元件合併模組中用到的字元特性有四個：最大字寬、最大字高、平均水平字距、平均垂直字距。首先，我們計算所有元件（總數量爲 n）的寬度與高度，放入序列 *pWidth* 與 *pHeight*，將兩序列由大而小方式排序後，利用取多數（85%）較大元件的平均寬度與高度的方式，得到目前的最大字寬 *MaxWidth* 與最大字高 *MaxHeight*：

$$MaxWidth = \frac{1}{0.85 \times n} \sum_{i=1}^{0.85 \times n} pWidth[i] \qquad (4)$$

及

$$MaxHeight = \frac{1}{0.85 \times n} \sum_{i=1}^{0.85 \times n} pHeight[i] 。 \qquad (5)$$

在平均字距的求法部分，則對每個元件找出距離最近的同行元件，再將之間的空隔（gap）大小訂爲兩元件間的字距。我們以求元件 *A* 的水平字距爲例，只考慮與元件 *A* 在 Y 軸部分有重疊且位於 *A* 的右邊之其他元件，找到距離 *A* 最近的元件 *B*，並將之間的空隔當作元件 *A* 與元件 *B* 的水平字距 $GapHor_A$。元件 *A* 的垂直字距，則考慮與元件 *A* 在 X 軸部分有重疊且位於 *A* 的下邊之其他元件，找到距離 *A* 最近的元件 *C*，並算出垂直字距 $GapVer_A$。如果 $GapHor_A \le GapVer_A$，表示元件 *A* 可能位在水平文字行上；如果 $GapVer_A < GapHor_A$，表示元件 *A* 可能位在垂直文字行上。將所有元件的水平與垂直字距放入序列 *pGapHor* 與 *pGapVer*，並判斷出每個元件是位在水平或垂直文字行上。如果位在水平文字行的元件數目大於位在垂直文字行的元件數目，表示此手寫稿文件是水平走向，反之則文件爲垂直走向。將 *pGapHor* 與 *pGapVer* 依大小排序後，如果是水平走向，我們可算出平均水平字距 $AvgGapHor=pGapHor[0.3 \times num_{hor}]$、與平均垂直字距 $AvgGapVer=pGapVer[0.1 \times num_{ver}]$，$num_{hor}$ 與 num_{ver} 表示在 *pGapHor* 與 *pGapVer* 序列內的元素個數。若是垂直走向文件則將兩係數 *0.3* 與 *0.1* 互換。

所有元件的字元特性（寬度、高度、水平間距、垂直間距）都求出，經過排序後，在其分佈統計中選擇動態的門檻值，當作所有元件的字元特性平均值。求出四種字元特性後，就可用來檢驗所有的元件，將元件間距夠小且合併後的區塊也滿足限制者予以合併。

2. 元件合併

對於每個元件 *A*，考量其他元件與元件 *A* 的合併可能性，假設有一元件 *B*，兩元件的水平間距爲 Gap_{hor}、垂直間距爲 Gap_{ver}，兩元件若合併，合併後寬度爲 *MrgWidth*、合併後高度爲 *MrgHeight*，合於下面條件者則表示 *A, B* 兩元件距離夠近，且合併後仍滿足條件限制，因此兩元件可合併：

$$(Gap_{hor} < AvgGapHor \;\; \&\& \;\; Gap_{ver} < AvgGapVer) \qquad (6)$$

及

$$(MrgWidht < MaxWidth \;\; \&\& \;\; MrgHeight < MaxHeight)。 \qquad (7)$$

這種以元件間隔爲基礎的合併方式並不具有遞移性，任何一元件可能找到多個相近的元件，都滿足合併的條件，不可以將這些元件一起合併，會導致合併後的大小超過限制。因此在此階段的合併，每個元件只可找滿足條件的最近元件合併。但是，此階段的合併也需作重複合併的動作，每次重複時，所有字元特性需重新計算，並重新考慮兩兩元件的合併可能性，直到沒有任何元件可再彼此合併爲止，會得到此第二階段的元件合併結果（如圖 5 所示）。

（三）第三階段元件合併

在前兩階段的元件合併中，我們可將重疊性高或者間距夠近的兩元件，予以合併。大部分手寫稿文件均可利用此兩階段的元件合併方式將手寫字元擷取出來。然而，手寫文字

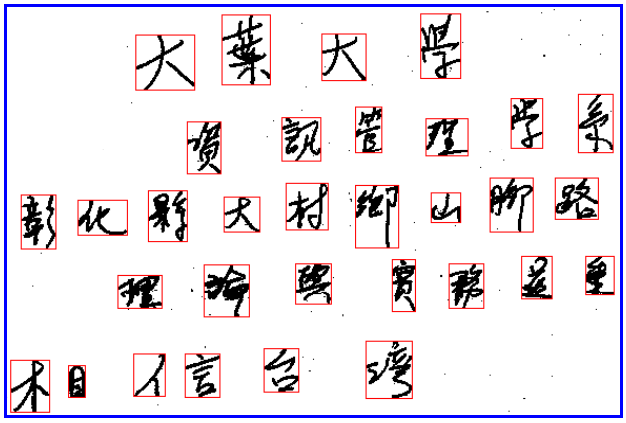

圖 5. 第二階段的元件合併結果

的變異性還是很大，在第二階段元件合併中，我們採用的條件是元件間距離需夠近，而且合併後的寬高不可超過限制。雖然所用的門檻值（threshold），均是依據手寫稿內容計算得到，是變動的而非固定。但是手寫的字元大小差異或間距變異過大時，就可能發生同字元的元件未合併，或者不同字元的元件合併了。因此，我們再提出第三階段的元件合併。

此階段的元件合併，主要是放寬合併後的寬高限制，主要利用元件前後間距變化度來決定是否合併。對於每個元件，找到四方向（左、右、上、下）最近的鄰居元件與相對的間距（Gap_{left}, Gap_{right}, Gap_{top}, Gap_{bottom}）。並算出兩組相反方向的元件間距比例：

$$GapRatio_{hor} = \frac{\max(Gap_{left}, Gap_{right})}{\min(Gap_{left}, Gap_{right})} \qquad (8)$$

and

$$GapRatio_{ver} = \frac{\max(Gap_{top}, Gap_{bottom})}{\min(Gap_{top}, Gap_{bottom})}。 \qquad (9)$$

假設元件 *A* 其左邊最近的鄰居爲元件 *B*（其間距爲上述的 Gap_{left}），同樣地算出假設合併後的寬度爲 *MrgWidth*，合於下面條件者則表示 *A*, *B* 兩元件距離夠近、兩側的間距差異過大、且合併後仍滿足條件限制，因此兩元件可合併：

$$MrgWidht < ThrWidth \ \ \& \ \ Gap_{left} < ThrGap \ \ \& \ \ GapRatio_{hor} > ThrRatio \qquad (10)$$

若考量的是右邊、上邊或下邊最接近的元件是否可合併，則將上述條件改爲如下相對的條件：

$$MrgWidht < ThrWidth \ \ \& \ \ Gap_{right} < ThrGap \ \ \& \ \ GapRatio_{hor} > ThrRatio \qquad (11)$$

$$MrgHeight < ThrHeight \ \ \& \ \ Gap_{top} < ThrGap \ \ \& \ \ GapRatio_{ver} > ThrRatio \qquad (12)$$

$$MrgHeight < ThrHeight \ \ \& \ \ Gap_{bottom} < ThrGap \ \ \& \ \ GapRatio_{ver} > ThrRatio \qquad (13)$$

其中 *ThrWidth*、*ThrHeight* 表示對合併後的寬高是否過大的條件限制，*ThrGap* 表示兩元件的間隔大小限制，*ThrRatio* 表示該元件與兩側鄰近元件的間距比例限制。

這種以元件兩側間隔差異度爲基礎的合併方式，雖然也設有寬高的限制，但爲了解決手寫字大小的變異度，寬高的條件較鬆，主要是考慮兩側間隔的差異度來決定是否合併兩元件。因此，此階段的合併也具有遞移性，兩兩元件利用計算間隔差異度偵測，記錄是否可合併。所有元件都檢視完畢後，即可將可合併的同組元件，一次合併。此階段的合併也需作重複合併的動作，每次重複時，所有限制條件必須重新計算，並重新考慮兩兩元件的合併可能性，直到沒有任何元件可再彼此合併爲止，會得到最後的單字擷取結果（如圖 6 所示）。

四、手寫文件讀序決定

在前節中，我們計算文件影像中所有元件的水平與垂直間距大小後，利用一般「字距比行距小」的特性，可得到每個元件可能位在水平或垂直文字行，統計後可得到該手寫稿文件是水平或垂直走向。根據一般人寫字的習慣，若是水平走向，文字行的讀序必定是由上而下，單字元的讀序則不定；

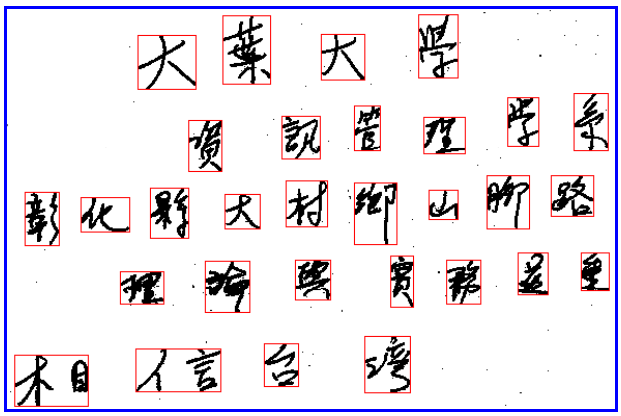

圖 6. 第三階段的元件合併結果

若是垂直走向，單字元的讀序必定是由上而下，文字行的讀序則不定。對於水平走向的單字元讀序，與垂直走向的文字行讀序，只能利用辨識後的前後字詞義判斷，才能確定讀序，因此不在我們的研究範圍。

假設文件判斷出是水平走向，以處理單字讀序爲由左到右方向爲例，該手寫稿文件的讀序，應該是將由上到下的每一水平文字行，由左到右讀出每個單字。我們首先將所有字元區塊對垂直位置（Y 軸座標）做排序，讓較垂直位置類似的區塊則由上到下排。再來，按照順序將每一個文字區塊 *A* 投映（mapping）到一塊與文件同寬的陣列 *pMap* 上，將 *pMap* 從區塊左邊界到右邊界位置都記錄成該區塊的編號，$pMap[i] = A$，$i = Left_A,..., Right_A$，$Left_A$ 與 $Right_A$ 表示文字區塊 *A* 的左右邊界。某個文字區塊 *K* 要投映到 *pMap* 之前，若發現它要投映的位置已記錄有其他區塊的編號了，則表示文字區塊 *K* 不屬於此文字行，跳過區塊 *K* 不予投映。當所有的文字區塊都看過後，投映在 *pMap* 上的區塊就是由上往下所看到的第一排字元（如圖 7 所示）。但因每行的字元數目不定，*pMap* 上所記錄的字元不一定在同一水平文字行，尚須以相鄰字元的垂直位置來判斷是否位在同一行。

首先，以所記錄字元中位置最高的字元 *I* 爲基準，先在 *pMap* 內往右找下一個字元 *I+1*，檢測其兩相鄰字元的垂直位置差異是否不大，表示屬於同一行文字，比較基準改爲字元 *I+1*；如果垂直位置差異過大，表示不屬同一行文字，則我們在字元 *I* 後面插入一屬性爲「空白」的新字元，再找下個字元是否與字元 *I* 位在同一行。若 *pMap* 內往右已無其他字元編號時，表示該行已看完，則最後插入一屬性爲「換行」的字元。往右看完後，同樣將基準回歸到字元 *I*，在 *pMap* 內往左找到相鄰且垂直位置差異不大的字元 *I-1, I-2*, …，如果垂直位置差異太大，仍然在該字元前插入一屬性爲「空白」的新字元。經過上述的步驟，我們已可確定哪些文字位在該行，並以空白字元與換行字元，表示該行文字的外觀。清空

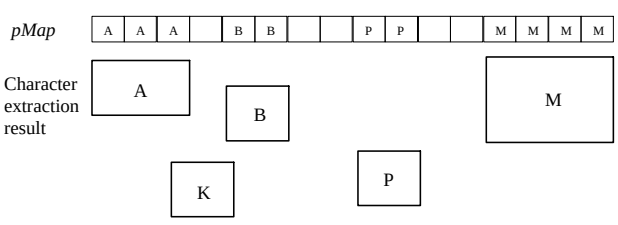

圖 7. 利用投映方式找出首行文字元

pMap 後，再以同樣的水平位置對映與垂直位置比對的方式，找到下一行文字。如此反覆的執行，即可將所有字元的順序都決定（如圖 8 所示）。

同樣的，對於垂直走向由右到左的手寫稿文件，我們先將文字區塊以 X 軸座標排序，將區塊位置投映到與文件同高的陣列上。以所記錄字元中位置最右邊的字元爲基準，分別往下與往上找到相鄰且水平位置差異不大的文字區塊，即可依順序找到同一垂直行的文字區塊。

五、實驗結果與分析

本文提出利用相連元件抽取、多階段元件合併與讀序決定方法，來開發手寫稿文件影像的文字抽取技術，是以 C++ 語言在 VisualStudio.NET 環境開發、在 Pentium 4-1.8G 硬體配備下測試。爲了驗證本文所提出的手寫文字抽取方法，我們請 15 人寫了 57 張手寫稿，合計有 1148 字。利用 Umax Astra 4800 型號之平台掃描器，以解析度 300 dpi（dots per inch）掃描成黑白影像，存成 tif 格式之影像檔。

對於每張手寫稿文件影像，分別利用相連元件抽取方式、只使用第一階段元件合併、只使用前兩階段元件合併、與完整使用三階段元件合併方式進行字元抽取，以人眼方式驗證抽取結果的正確性。我們以抽取率來比較各方式的字元抽取結果，所謂抽取率（extraction rate）定義爲所有文字中，有多少文字被完整擷取出來。若某文字的部首未完全合併，則算該字抽取錯誤；若有某文字與鄰近文字的部首元件合併了，則兩文字都算抽取錯誤，表 1 是我們的測試結果。

從表 1 所顯示的手寫文字抽取結果，我們知道有少部分中文字（如‘一’），本身是一筆畫可以寫完的，並不會有元

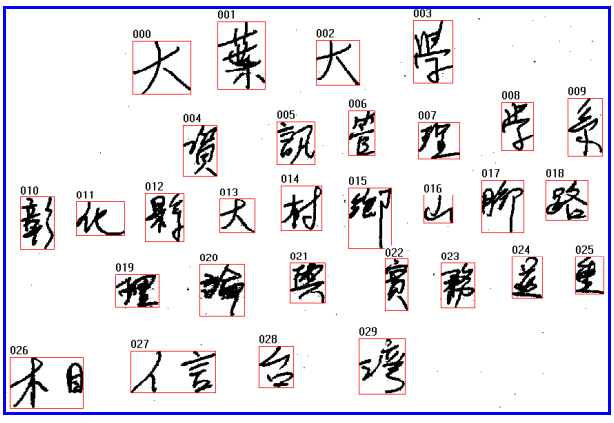

圖 8. 水平手寫稿文字擷取與讀序決定後結果

表 1. 手寫稿文件之文字抽取結果

	Connected components	First one stage component merging	First two stages component merging	Three stages component merging
Extraction rate	13.06%	65.33%	95.82%	98.43%

件需合併的問題。因此，即使未用任何階段的元件合併，仍然可將這些字元完整抽取出來。有些中文字並非單一部首，但是部首間有相當大的重疊性（如‘回’），則使用第一階段的元件合併，只依據元件間的重疊性，即可將很多的多部首文字予以合併成完整字元。仍然有些文字有幾個部首組成，但是部首間的重疊性不高，甚或有些間距，大部分人的手寫字，同一字的部首間距不致太大，各字元的寬高變異度也還可接受，則再利用第二階段元件合併，考慮元件間距離與合併後寬高的限制，一樣可以將其他尚未合併的同字部首，予以合併。經過前兩個階段的元件合併，其實幾乎所有同字部首均已完成合併。爲處理手寫變異度較大的文件，第三階段的元件合併，會將合併限制更爲放鬆，主要考量元件與兩側鄰居元件的間距差異比例，來將一些字元部首合併後可能寬高過大，但是部首間距離與字元間距離還是有明顯差異者，予以合併，可再提昇部分文字抽取結果。

由於切字的三大類方法（dissection method, holistic method, recognition-based method），所著重處理的問題各有不同。本研究所提出的方法歸屬於 dissection method 類別。與另外兩類切割方法的比較上，holistic method 著重無法切割或不容易切割的相連字（如 fi, rn）辨識，以歐美語系爲主，不適用於中文字。recognition-based method 的類別，切字效果好壞與使用的辨識核心有很直接的關係。只用辨識核心 [24] 來修正切字效果，如果不使用與 dissection method 相關的一些字元特性（如字元大小、字元間距），切字效果的提昇其實有限。若同時考慮辨識結果與本論文提出之字元特性，效果會較好，但是需有額外的辨識時間花費。表 2 是我們的測試結果。

分析一些字元無法完整被擷取出來的主要原因，大概有三類。第一類是由於個人的手寫習慣，使得有些手寫字彼此接觸（如圖 9 (a)），因爲我們認識這些字，以人的觀點還是看得出每個字的實際範圍。但是，利用系統抽取出的相連元件就已經包含兩字元的部首，不管用何種合併方式，仍然無

表 2. 所提方法與辨識基礎方法之效果比較

	Proposed method	Recognition-based method	Combined method
Extraction rate	98.43%	86.16%	98.61%

(a)　(b)　(c)

圖 9. 可能錯誤情況：(a) 兩字元碰觸；(b) 部首間距比字元間距大；(c) 標點符號夾在字句中

法將原以相連的字元分割。要解決此問題，必須利用分析各元件的外型，找出可能的切割位置，再利用辨識核心作驗證，即可確定何切割位置最恰當。第二類錯誤原因，是字元內部首的間距大於兩字元的距離（如圖 9 (b)）。再利用第二或第三階段元件合併時，反而會將不同字元的部首元件合併。要利用相連元件間的空間關係，將同字元的元件合併，其實不太容易。解決之法，還是同前類問題，可能需利用辨識結果，來判斷較準確的切割位置。第三類問題是一般人在手寫文字時，常常習慣性會有點標的動作（如圖 9 (c)），也就是會在寫完幾字後用筆點一下紙，有時會寫出逗點或句點等標點符號。如果標號的尺寸較小，可能會在我們的雜訊去除階段即會清除，不會影響字元抽取結果。如果標號尺寸較大，不會被當作是雜訊，但是相對的其尺寸與其他字元比起來，還是較小，會被當作某字元的一部份而被合併。要處理此問題，可利用標點符號辨識核心，先對所有小元件進行符號辨識，如果確定是某符號（信心度夠高），可先將此符號從手寫稿文件中移除，以免影響後續字元抽取的結果。但是，有時會將一些字元內的小點誤認爲逗點或頓號，反而導致該字元有些元件被移除而不完整。

上述的說法似乎以辨識爲基礎的文字抽取效果較好？文字辨識其實較花時間，如果利用文字辨識來確定文字位置，可能需要執行多次可能字元的辨識，才能確定分割位置，會導致整個執行時間較緩慢。以我們提出的相連元件與多階段元件合併技術，會將文字抽取部分的時間加速，縱使有些不完整的文字抽取結果，一般的文字辨識核心還是可以成功辨認字元的小部分不完整。若不完整的區域太大，我們可再利用辨識核心對一些辨識信心度過低的文字，重新考慮

該字元是否要重新切割，或者與相鄰文字合併成提同一文字，可得到較好整體辨識效果。

中文、英文與數字的特性各有不同，對於中英數字混合的文件，在文字辨識系統之文字抽取、文字辨識、辭典校正等模組，都會造成處理上的困難。在本文探討的研究方向，主要以中文字的文字抽取爲主。對於手寫數字與手寫英文字，如果與手寫稿中其他中文字有類似的特性（如字元寬度與高度、水平與垂直字距），以本文提及之方法，一樣可以正確將文字位置擷取。如果手寫英數字與中文字有明顯不同的特性，例如「1I1」三字元間距太小（相對於其他中文字的字距），被合併成「川」的機會也相對大。對於中英數字夾雜的印刷文字抽取，還是需配合辨識核心，會有較好的效果。但是對於中英數字夾雜的手寫稿，因爲變異度相對較大，即使利用辨識核心，手寫辨識的信心度必須夠穩定，否則對文字抽取效果的加強還是有限。另外，也可配合辭典校正的後處理方式，利用修正辨識結果來調整正確的文字切割位置。

六、結論

一個實用的光學手寫字擷取模組在此研究論文中被說明。一張淡色紙、深色字的手寫稿經掃描輸入後，先做相連元件偵測，得到一些像素相連的元件區塊，並將一些夠小的元件視爲雜訊予以去除。而後利用元件間彼此的重疊性，進行第一階段元件合併；統計一些字元特性（字寬、字高、字距等）後，利用元件間的間隔大小，來進行第二階段元件合併。然後，第三階段的元件合併，則利用元件與左右或上下兩側鄰居元件的距離差異性，來決定是否合併。擷取出所有字元後，最後再利用字元區塊投映與鄰近元件位置差異檢測方式，決定出所有字元的讀序。此手寫字擷取模組具有實際的應用性，如果日後搭配文字與符號辨識核心，其文字擷取效果與日後應用範圍將可更提昇。

參考文獻

1. Bin, Y. and A. K. Jain (1996) A robust and fast skew detection algorithm for generic documents. *Pattern Recognition*, 29(10), 1599-1629.
2. Casey, R. G. and E. Lecolinet (1996) A survey of methods and strategies in character segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7), 690-706.
3. Chen, J. L. and H. J. Lee (1998) An efficient algorithm for form structure extraction using strip projection. *Pattern Recognition*, 31(9), 1353-1368.
4. Chen, X. N. and D. C. Tseng (1995) Form-structure extraction for table-form recognition. Proceedings of IAPR Conference on Computer Vision, Graphics and Image Processing, Taoyuan, Taiwan.
5. Cheng, F. H. and W. H. Hsu (1991) Research on Chinese OCR in Taiwan. *International Journal of Pattern Recognition and Artificial Intelligence*, 5(1&2), 139-164.
6. Fan, K. C. and L. S. Wang (1998) Classification of machine-printed and handwritten texts using character block layout variance. *Pattern Recognition*, 31(9), 1275-1284.
7. Gatos, B., N. Papamarkos and C. Chamzas (1997) Skew detection and text line position determination in digitized documents. *Pattern Recognition*, 30(9), 1505-1519.
8. Gonzalez, R. C. and R. E. Woods (2002) Noise reduction. *Digital Image Processing*, 2nd Ed., 243-253. Prentice-Hall, New Jersey, NJ.
9. Govindan, V. K. and A. P. Shivaprasad (1990) Chinese recognition - a review. *Pattern Recognition*, 23(7), 671-683.
10. Hirayama, Y. (1993) A block segmentation method for document images with complicated column structures. Proceedings of 2nd International Conference on Document Analysis and Recognition, Tsukuba Science City, Japan.
11. Ho, C. T. and L. H. Chen (1996) A High-speed algorithm for line detection. *Pattern Recognition Letters*, 17, 467-473.
12. Jiang, H. F., C. C. Han and K. C. Fan (1997) A fast approach to the detection and correction of skew documents. *Pattern Recognition Letters*, 18(7), 675-686.
13. Lee, S. W., D. J. Lee and H. S. Park (1996) A new methodology for gray-scale character segmentation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10), 1045-1050.
14. Lin, C. F., Y. F. Fang and Y. T. Juang (2001) Chinese text distinction and font identification by recognizing most frequently used characters. *Image and Vision Computing*, 19, 329-338.
15. Lu, Y. (1995) Machine printed character segmentation - an overview. *Pattern Recognition*, 28(1), 67-80.

16. Lu, Y. and M. Shridhar (1996) Character segmentation in handwritten words - an overview. *Pattern Recognition*, 29(1), 77-96.
17. Mori, S., K. Yamamoto and M. Yasuda (1984) Research on machine recognition of handcrafted characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(4), 386-405.
18. Nagy, G. (1988) Chinese character recognition: a twenty-five-year retrospective. Proceedings of 9th International Conference on Pattern Recognition, Rome, Italy.
19. Okamoto, M. and M. Takahashi (1993) A hybrid page segmentation method. Proceedings of 2nd International Conference on Document Analysis and Recognition, Tsukuba Science City, Japan.
20. Otsu, N. (1979) A thresholding selection method from gray-scale histogram. *IEEE Transactions on Systems, Man, and Cybernetics*, 9, 62-66.
21. Pal, U. and B. B. Chaudhuri (1996) An improved document skew angle estimation technique. *Pattern Recognition Letters*, 17(8), 899-904.
22. Panjwani, D. K. and G. Healey (1995) Markov random field models for unsupervised segmentation of textured color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(10), 939-954.
23. Tseng, L. Y. and R. C. Chen (1998) Segmenting handwritten Chinese characters based on heuristic merging of stroke bounding boxes and dynamic programming. *Pattern Recognition Letters*, 19(10)**,** 963-973.
24. Tseng, Y. H., C. C. Kuo and H. J. Lee (1998) Speeding-up Chinese character recognition in an automatic reading system. *Pattern Recognition*, 31(11), 1601-1612.
25. Tseng, Y. H., C. C. Kuo and H. J. Lee (1998) Typeface identification for printed Chinese characters. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(2)**,** 173-190.
26. Tseng, Y. H. and H. J. Lee (1999) Recognition-based character segmentation using probabilistic Vitervi algorithm. *Pattern Recognition Letters*, 20(8), 791-806.
27. Wang, J. and J. Jean (1994) Segmentation of merged characters by neural networks and shortest path. *Pattern Recognition*, 27(5), 649-658.

收件：93.06.08　修正：93.08.25　接受：93.09.06