

# 國立交通大學

資訊學院 資訊學程

碩士論文

以筆畫線段為基礎的  
手寫中文字旋轉偵測

Rotation Detection of Handwriting Chinese Character

Based on Strokes Segments

研究生：陳柏煜

指導教授：李素瑛 教授

中華民國九十九年六月

# 以筆畫線段為基礎的 手寫中文字旋轉偵測

Rotation Detection of Handwriting Chinese Character

Based on Strokes Segments

研 究 生：陳柏煜

Student : Pao-Yu Chen

指 導 教 授：李素瑛

Advisor : Dr. Suh-Yin Lee



**Submitted to College of Computer Science**

**National Chiao Tung University**

**in partial Fulfillment of the Requirements**

**for the Degree of Master of Science**

**in**

**Computer Science**

**June 2010**

**Hsinchu, Taiwan, Republic of China**

**中華民國九十九年六月**

# 以筆畫線段為基礎的 手寫中文字旋轉偵測

學生：陳柏煜

指導教授：李素瑛博士

國立交通大學

資訊學院

資訊學程碩士班

## 摘 要

隨著平板電腦、手持式行動裝置的日漸普及，手寫文字辨識的需求也日益增加，這樣的需求在使用漢字的地區更加迫切。當漢字手寫識別技術的應用日趨成熟時，建構一個自由自在的書寫輸入環境的需求也就慢慢浮現出來。本研究的目的，是希望在進行手寫識別之前，將輸入軌跡的旋轉角度找出來加以校正，讓使用者不需要把輸入裝置面向自身的方向來書寫，也能夠得到正確的識別結果。本研究中使用漢字筆畫線段中的基本特性，由左至右的水平筆畫以及由上至下的垂直筆畫，它們的垂直關係是不會隨著軌跡的旋轉而變化的，只要確立了這兩種筆畫目前的角度，旋轉角度也就可以偵測出來。本論文的實驗，首先要找出輸入軌跡的筆畫線段的方向與長度，其次使用兩種不同的方法，筆畫線段投票法搭配經驗法則與 K-mean clustering 的統計方法，找出水平與垂直筆畫目前的角度，比較兩者得到的結果的優劣以及在處理速度上的差異。另外引入兩個輔助的實驗，其一是驗證手寫識別程序傾斜軌跡處理效果，以證明本實驗的方法是真正有效的；另一則是針對輸入軌跡使用 PCA 方法嘗試找出旋轉角度做為本論文方法的對比。最後，從兩個偵測方法中找出一個較佳的方法來建構一個手寫中文字旋轉偵測系統。

# Rotation Detection of Handwriting Chinese Character Based on Strokes Segments

Student : Pao-Yu Chen Advisor : Dr. Suh-Yin Lee

## ABSTRACT

As TabletPCs and Handheld devices are widely sepreaded, handwriting characters recognition systems are requested, and the request is more important in areas using Chinese Characters. More and more stablized applications of Handwriting recognition technology makes the requirement. A more free handwriting input environment is prompted. This research focuses on rotation angle detection during the preprocess of handwriting characters. Users need not turn the input device to themselves before starting input. Based on the basic characteristics of input strokes, segments of Chinese characters, horizontal strokes from left to right and vertical strokes from top to bottom, this research can detect the rotation angle if such invariant relations are found. First of all is to compute the direction and length of each stroke segment of the input writing. Secondly, two different methods and their combination, structure analysis with heuristic rules and K-mean clustering, are used to detect the rotation angle. Two associated sets of experiments are performed. One is verifying skew/rotation limitation of the handwriting recognition engine to make sure the research is working. And the other is applying Principal Component Analysis on input writing to detect the rotation angle as contrast. Finally, comparing these two results and pick the better one to construct a Handwriting Chinese Characters Rotation Detection System.

## 誌 謝

不才承蒙恩師 李素瑛教授不棄，惠予指導，之後經歷改換研究方向，在很長一段時間還陷入低潮，論文撰寫的時候經常會陷入思考的迷宮，但是老師最後還是給了機會讓我改進，經常的開導點撥，引領著我走出迷惘。由於老師的諄諄教誨以及耳提面命，如今才得以完成論文撰寫，師恩深重，點滴謹記在心，不敢或忘，謹在此由衷致上萬分感謝。

能夠有這個機會再次回到學校進修，要感謝的是公司的主管 蔡義泰博士以及 洪成安博士，不僅在入學審查的時候給予推薦，在工作安排上給予方便，在百忙之中還撥冗提供許多寶貴的建議，使資料查閱與論文的撰寫得以快速推進。

我的同學林俊男，再次一起走過一段求學的時間，不僅在尋找指導教授的時候大力幫忙，在最後論文撰寫的時候也提供自身的經驗以及手稿作為參考，非常感謝他的協助。

妻子秋樺在背後的支持，鼓勵我重回學校進修，擔負處理家中大小瑣事的責任，使我得以致力在學業以及最後論文的撰寫，實在是功不可沒，在此特表示謝意於萬一。

還有許多人在修業的一路上給予許多的幫助，從入學、選課、到最後論文完成，限於篇幅無法一一詳列，特別在此一併表達感謝之意。

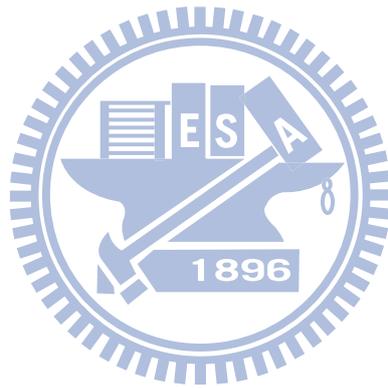
除了方向、方法上的提點以及原始資料的收集以外，恩師、公司主管以及其他協助者，並未干涉本論文中的實驗以及結論，如果文中有錯漏、謬誤、悖論之處，概由本人擔負應有之責任。

# 目 錄

	頁次
中文摘要 .....	i
英文摘要 .....	ii
誌謝 .....	iii
目錄 .....	iv
表目錄 .....	v
圖目錄 .....	vi
第一章、 緒論 .....	1
1.1 背景介紹 .....	1
1.2 研究動機與目的 .....	1
1.3 組織架構 .....	2
第二章、 相關領域及研究 .....	3
2.1 光學字元辨識 .....	3
2.2 線上手寫識別系統 .....	5
第三章、 研究方法與系統架構 .....	9
3.1 研究方法 .....	9
3.2 系統架構 .....	18
第四章、 實驗結果 .....	23
4.1 手寫識別核心的傾斜忍受能力 .....	24
4.2 線段分群的實驗-筆畫線段投票法 .....	26
4.3 使用不同方法偵測旋轉角度-PCA .....	29
4.4 改善線段分群的表現的實驗 .....	36
4.5 兩種線段分群方法處理速度比較 .....	39
4.6 實驗系統的成果展示 .....	41
第五章、 實驗結果討論與未來研究方向 .....	44
5.1 實驗結果討論與分析 .....	44
5.2 未來研究方向 .....	45
第六章、 參考文獻 .....	47

## 表目錄

表 4-1-1	手寫識別核心對於逆時針方向的傾斜容錯能力 調查 .....	24
表 4-2-1	32 等分的旋轉角度偵測後的校正結果識別率 ...	27
表 4-2-2	64 等分的旋轉角度偵測後的校正結果識別率 ...	27
表 4-2-3	128 等分的旋轉角度偵測後的校正結果識別率 ...	27
表 4-2-4	比較未抹平與抹平後的軌跡對平均識別率的 影響 .....	29
表 4-4-1	使用 K-mean clustering 對偵測旋轉角度後 的前十位累計辨識率 .....	38
表 4-5-1	筆畫線段投票法與 K-mean clustering 處理速度 的比較表(單位:ms) .....	39



## 圖目錄

圖 2-1-1(a)(b)(c)	以塗抹方法求取傾斜角度示意圖	5
圖 2-2-1	漢字書寫起點示意圖	7
圖 2-2-2	推論求取文字行走向方法示意圖	8
圖 3-1-1	漢字水平與垂直筆畫分解圖	9
圖 3-1-2	複合筆畫例圖	10
圖 3-1-3	以筆畫方向與組成分類的漢字例圖	10
圖 3-1-4(a)(b)(c)	筆畫方向與線段方向示意圖	11
圖 3-1-5(a)(b)	字體內部傾斜範例	12
圖 3-1-6(a)(b)	字體外部傾斜範例	12
圖 3-1-7(a)(b)(c)(d)	同時具有外部與內部傾斜的手寫字 範例	13
圖 3-1-8(a)(b)(c)(d)	漢字元件拆分範例	14
圖 3-1-9	本研究系統定位圖	15
圖 3-1-10(a)(b)(c)(d)	橫豎筆畫相對關係示意圖	16
圖 3-1-11(a)(b)(c)	經驗法則算法的實例運用示意圖	17
圖 3-2-1(a)(b)	方向畫分與量化示意圖	18
圖 3-2-2	量化方向表達示意圖	19
圖 3-2-3	由 major direction M 找尋 second major direction S	20
圖 3-2-4	中文書寫特性測試示意圖	21
圖 4-1-1	「璇」的傾斜角度變化示意圖	25
圖 4-2-1	軌跡移除雜訊示意圖	28
圖 4-3-1(a)(b)(c)(d)	沒有旋轉的樣本經過 PCA 運算的 部分結果	33
圖 4-3-2(a)(b)(c)(d)	旋轉 $112.5^\circ$ 後的樣本經過 PCA 運算 後的部分結果	34
圖 4-3-3(a)(b)(c)(d)	PCA 方法成功偵測旋轉角度案例圖	35
圖 4-4-1(a)(b)(c)(d)	有許多無法合併的小線段的 手寫字	36
圖 4-6-1	實驗系統成果展示一號圖	41

圖 4-6-2	實驗系統成果展示二號圖	.....	41
圖 4-6-3	實驗系統成果展示三號圖	.....	42
圖 4-6-4	實驗系統成果展示四號圖	.....	42
圖 4-6-5	實驗系統成果展示五號圖	.....	43
圖 5-1-1(a)(b)(c)(d)	缺乏橫筆或者豎筆的漢字範例	...	44



# 第一章 緒論

## 1.1 背景介紹

將各式的資料數位化是資訊時代的重要課題，眾多的輔助輸入工具與程式也陸續不斷的出現，鍵盤是第一個被廣泛接受且目前還是必要的輸入工具，然後出現了光學識別系統，包含文字識別、圖像識別、條碼掃描、人臉識別等等，特別是光學文字識別 (Optical Character Recognition, OCR) 對於大批量的文字資訊的數位化有著重大貢獻，然後是線上的手寫識別系統 (Online Handwriting Character Recognition, OLHCR) 以及語音識別系統。對於大多數有著小量輸入資料的需要卻不熟悉鍵盤輸入法的使用者來說，手寫識別系統是他們優先的替代選擇，因為其學習週期短，受裝置的限制少，成熟的技術使得辨識率高達九成五以上，受歡迎的程度從日漸增加而且不斷加速遞增的應用可見一斑。技術的改進，讓使用者從原本的必須工整順序書寫輸入資料，到可以連筆順序書寫，現在則是可以用連筆且不按照順序書寫；字體從原來不可歪斜，改進到容許一定程度的歪斜。

但是使用者或者是因為習慣，或者未受訓練，或者受環境限制，在書寫時還是可能會歪斜超過辨識系統的容許程度，甚至是旋轉或者倒轉，這些字就無法被辨識出來。已經有研究者注意到線上手寫字傾斜角過大的這個現象，並且嘗試加以解決[2]。

## 1.2 研究動機與目的

線上手寫字的書寫方向與每次輸入的字數都相當的隨意，不像光學字元識別 (OCR) 針對掃描輸入的文件影像，有大量整齊排列的字元影像以及行的資訊或者輔助的框線可供判斷傾斜或者旋轉的方向。

利用硬體配備[4]，例如陀螺儀 (Gyroscope)，來偵測書寫裝置的傾斜或是旋轉角度是一個簡單暴力的方式，但是這樣做的代價是裝置

成本的提高，而且對於像是書寫裝置本身沒有傾斜或者旋轉，而是使用者本身換一個方向來書寫的這種現象，一樣無能為力。

按照人類辨識一個字的習慣，一般會進行好幾次的旋轉比對，如果正好符合印象中的結構，那麼我們就會說「辨識出這個字」。以這樣的認知為基礎，我們可以把一個字分別旋轉不同的角度後進行辨認，然後取與預先訓練儲存的樣板差距最小字的作為最後的辨識結果。採用這種方法首先必須要有高效率的辨識核心，如果必須對不同的旋轉角度建立與儲存模板，則另外還需花費倍數的時間與儲存體才能夠實行。

有沒有其他的方法可以偵測出手寫字傾斜旋轉的角度，卻不需要花費倍數的儲存資源以及運算時間呢？以此為條件，本論文的目的希望能夠找到有效的方法來偵測出所要識別的漢字的傾斜與旋轉的角度，並且在偵測過程中不需要進行識別來輔助確認所偵測到的旋轉角度，只有在偵測出旋轉角度之後才會進行辨識字元的步驟，這樣的方法就可以推廣到任何中文手寫的應用產品，例如 PDA(Personal Digital Assistant)、行動電話、以及 SoC (System on Chip) 等，提供使用者更方便的數位環境。

### 1.3 組織架構

本論文之章節組織編排如下：

第二章相關領域及研究，了解目前在手寫字的傾斜與旋轉偵測上的方法及其成果。第三章系統架構，說明本論文核心是利用的漢字特徵來完成角度的偵測，並且運用經驗法則輔助偵測以架構整個系統。第四章實驗成果，根據我們架構的模型運用在測試資料以及實際書寫的統計資料。第五章實驗結果討論及未來研究方向，探討造成方法失效的原因，說明研究心得以及成果展示，並指出未來研究上可以改進的項目。第六章參考文獻是本論文參考引用的所有相關文獻。

## 第二章 相關領域及研究

本節將探討在光學字元辨識、線上手寫辨識等領域之中，關於傾斜與旋轉角度的偵測的方法，了解其具體的做法以及優缺點，並且論述這些方法與本論文方法對於應用要求上的差異性。

### 2.1 光學字元辨識

光學字元辨識(Optical Character Recognition, OCR)是第一個提供自動書面資料數位化的工具，自從 60 年代起，經過將近五十年的發展，這項技術已經十分的成熟，而線上手寫字元識別(Online Character Recognition, OLCR)也是在這個時期開始發展。如果不考慮線上手寫系統所具有的筆畫次序的資訊，使用者單純從影像上來說這兩者是一樣的，所以兩者的技術有很多可以共通使用。

按照一般人對字元辨識的習慣，無法立即識別的字或者是無法用已知的規則拆解組合所要識別的字元的時候，我們會嘗試進行小角度的傾斜或者旋轉，以期找到符合預設印象的字元。所以對於線上手寫字元的傾斜與旋轉偵測，先把軌跡轉換成為二進位的影像，再利用 OCR 的傾斜偵測方法[3]進行校正是一個很直觀的解決方式。

這個方法首先針對影像的橫軸(X 方向)以及縱軸(Y 方向)進行投影，沿著行進方向，計算出每一組連續黑點(Run)的長度，並記錄在這個 Run 的最後一個黑點上，由這些數字組合成兩個特徵影像(Burst Image)，以大幅減少所需要處理的影像資料量，再對這兩個影像中的每一個黑點(非 0 的點)進行由(X, Y)到( $\rho$ ,  $\theta$ )的霍夫轉換(Hough Transform,  $\rho = X*\sin\theta + Y*\cos\theta$ )，對每一個 $\theta$ 做完之後的結果累計在( $\rho$ ,  $\theta$ )的二維陣列之中，轉換後的結果之中累計次數最高的值，就是在影像中所要找尋的文字行裡面的底線，針對這個找出來的底線進行傾斜角度的偵測。為了節省計算量，傾斜的角度被限制在 15° 以下，即便如此，文中提到處理一張文字影像還是需要 67 秒的時間，顯然這是不符合我們對於處理速度必須要近似於即

時(RealTime)的要求，對於偵測角度的限制也不是我們所能接受的，且線上手寫字通常沒有行與列的資訊，每次調用識別核心的時候需要識別的字數也不多，在資源有限的嵌入式系統通常是一個，而資源較多、可以使用程式自動切字的桌上系統則大約是 20 個左右，因此這個頗受歡迎的方法不適合用來解決我們的問題。

另外一個常用的方法就是利用塗抹(Smear)的方式來找出文字行的傾斜角度[5][6]。首先將影像縮小到原來的 1/4，以此作為後面偵測之用的影像；而文字在影像中通常以黑點來表示，所以我們在影像中找尋連續的黑點，稱之為 Run，如果這個 Run 長度滿足所設定的門檻值(threshold)的話，就可能是文字的所在，當然也有可能是圖形，不過目前先不考慮圖形的因素。如果沿著這個 Run 在一小段白點(空白)之後，又可以找到一個足夠長的 Run，顯然這就是一個文字行，那麼我們就把這一小段白點也全部變為黑點，讓整個文字行連起來變成一個黑色的區塊，這個過程稱為「塗抹」。接著從這個黑色區塊的中找兩個相距足夠遠的垂直線(我們目前針對水平文字行，所以找的是垂直線)，這兩個垂直線的第一個黑點的 Y 軸方向的差，以及這兩個垂直線的距離，就可以讓我們求出整個文字行的傾斜角度。

這個方法有以下兩個優點，首先是簡單，因為不需要對影像作縮小與塗抹以外的處理；其次是計算量很小，因為我們只處理原始影像的 1/4 的資訊量，如果縮小之後的影像其文字行還是可以被清晰的分辨出來，1/16 或者是 1/64 的縮小影像可以進一步減少計算量。但是他的缺點也是很明顯的，就是傾斜的角度不可過大，以免塗抹的時候發生跨行塗抹的情況，大約 $\pm 5^\circ$ 之間是可以正常運作的；文字行必須足夠長，不然所取的資訊可能沒有足夠的代表性，甚至找不到兩個相距足夠遠的垂直線。因此這個方法也不符合我們手寫字傾斜旋轉偵測的需要，雖然他的速度比前面所提到的方法[3]要快的多，但是其他的限制更多。圖 2-1-1(a)(b)(c)以塗抹方法求取傾斜角度示意圖，分別表示沒有經過塗抹的原始圖、利用 Run-length 塗抹過後的結果、以及塗抹過後找尋傾斜角度的方法，只要圖中的

兩個有圓圈記號的地方距離足夠遠，而且角度不是很大，求取出來的結果基本上是可以信賴的。

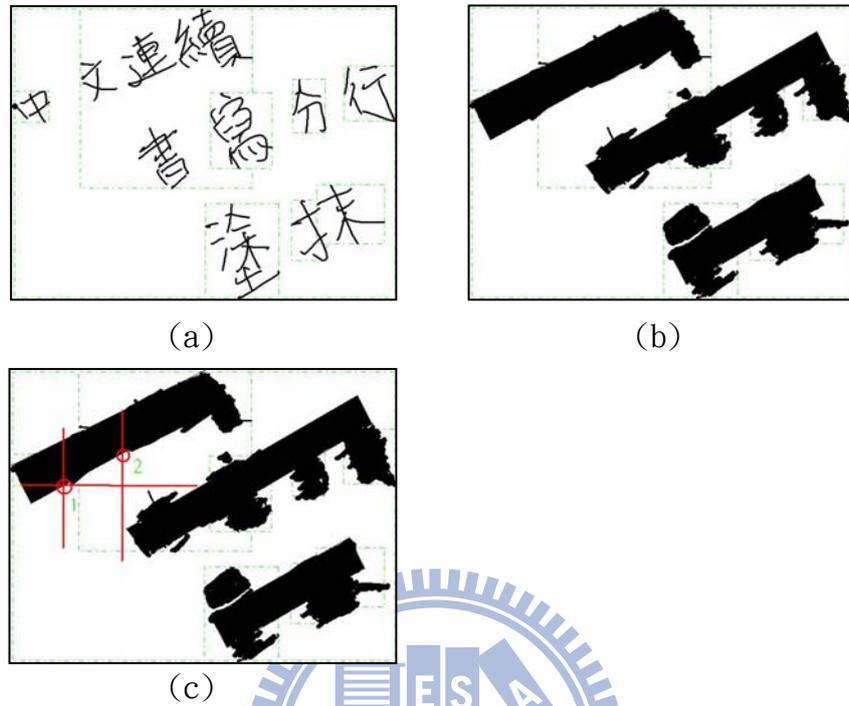


圖 2-1-1(a)(b)(c). 以塗抹方法求取傾斜角度示意圖

## 2.2 線上手寫識別系統

線上手寫識別系統 (Online Handwriting Character Recognition, OLHCR) 與光學識別系統在同一個時期發展，早期的手寫系統使用的是與 OCR 一樣的做法，也就是說 OCR 是針對很多個字的影像，而 OLHCR 則是針對少量文字的影像。但是在 OCR 容易解決的問題，到 OLHCR 變得複雜起來，因為人寫的字無法如同機器印刷一樣的精確，即使是同一個人連續兩次寫的同一個字，也可能存在著差異；個人的書寫習慣的不同，則造成同一個字由不同的人寫出的時候可能會有很大的不同，例如字體的間距、每個元件(Radical)的比例，等等的差異性就如同 OCR 所遭遇到的不同字體的問題。OCR 可以藉著訓練不同的字體的數據庫(database)來解決字體的差異問題，畢竟常用的字體的數據是已經知道的資訊，不會讓 database 無限制增長下

去。但是 OLHCR 若是要針對每個人的書寫習慣來建立專屬的數據庫，那個數量實在太龐大了，無法當成實用的產品。所以首先必須找出不同的人手寫字的相同之處，以這些相同之處當成特徵，可以有效的解決使用者因素所造成的問題，例如：拜教育普及之賜，我們有了所謂的「標準筆順」以及「標準字體」，規定每一個字的筆畫書數、書寫的筆畫次序、筆畫的位置、組字元件或部首的之間的大小比例、以及行筆的方向(從上至下，由左至右)等等，只要是在學校學過書法的使用者，大多會遵循這些規則，這就是 OLHCR 可以應用的特徵。

手寫識別依照樣本特徵的表示方式，可以大致分為三類：結構表示法、結構統計表示法、統計表示法[12]。結構表示法利用漢字的結構性特徵，例如筆畫的端點以及行筆方向、筆畫之間的關係、以及字的部首和衍生過來的共用元件(通稱為 radical)等等串連起來建立字的模板以及進行輸入軌跡的比對；而結構統計法不僅是利用漢字的結構資訊，更進一步的用統計的方式將筆畫結構的屬性關係的變異與平均數轉換成為 probability density functions(PDFs)，除了可以避免之前完全使用結構性描述的時候需要的人工編程來記錄這些特性之外，辨識結果也更加的穩定；最後的統計表示法則不再需要是先建立漢字的結構性描述，而是將軌跡轉換成為二維影像之後，利用類神經網路進行特徵的抽取之後建立模板，使用者輸入的軌跡也是用同樣的方式抽取特徵之後，再與已經建立的模板進行比對，最後識別出所寫的字。

結構法以及統計結構法既然要用字的結構來建立模板，所以在編程的時候就必須把傾斜的資訊當作變異的資訊輸入，之後才能夠處理傾斜的情況，如果有些元件的結構敘述沒有建立適當的變異資訊，有傾斜現象發生的時候其識別結果就會產生錯誤，不僅編程的時候要花費大量的時間精神，對於傾斜書寫的字元的識別穩定性也比較差。統計表示法處理傾斜書寫字元的方式則因為方法的特性而有比較好的改進，收集傾斜的軌跡之後不需要進行人工編程的修改，只要傾斜的軌跡足夠多，系統抽取出來的特徵就會包含傾斜的變異，在處理傾

斜字元的時候其辨識穩定性也比較高，但是對於字元的傾斜比較嚴重以及旋轉的情況還是無法獲得有效的處理，吸收太多的變異的話甚至可能會產生排拒正確輸入的情況。

對於這種傾斜嚴重以及旋轉的情況，在前處理的時候先行根據字本身[14]、短語(word)[13][15]、書寫筆劃以及文字行[11]的幾何特性以及書寫習慣進行校正，以期得到正確的識別結果，或者減少核心需要關注的訓練樣本的數量。漢字的書寫習慣一般是由上至下，由左至右，根據統計[14]，大部分的中文字來說，起點都是在左上角，所以對於任何輸入的軌跡，第一個步驟就是將整個字旋轉，使得它的第一筆劃的起點在左上角，如圖 2-2-1 漢字書寫起點示意圖所示。右側黑色字體表示使用者書寫的樣子，左側紅色字體表示程式校正之後的結果，大的圓點表示第一畫的起筆位置。

這個假設相當的粗略，很多漢字的第一畫的起筆位置並不在左上角，實驗結果顯示這個假設在某些角度的時候特別有效，有高達九成的校正成果，當然另外一些角度則與不校正的結果是一樣的，平均可以有 52% 的改善，因此，必須進行第二個步驟以提升整體表現。再次觀察旋轉後的資料並且比較與統計那些無法被改善的字的真正的角度與校正的角度，發現它們即使是被校正錯誤，也會落在固定的幾個角度， $0^\circ$  表示是正確被校正的樣本， $\pm 45^\circ$  表示這些樣本校正後的誤差角度，既然樣本的數量已經獲得控制，那麼就把這些樣本也納入識別核心的訓練樣本之中，利用辨識的方式來處理這樣的誤差，就可以得到可以接受的識別率。

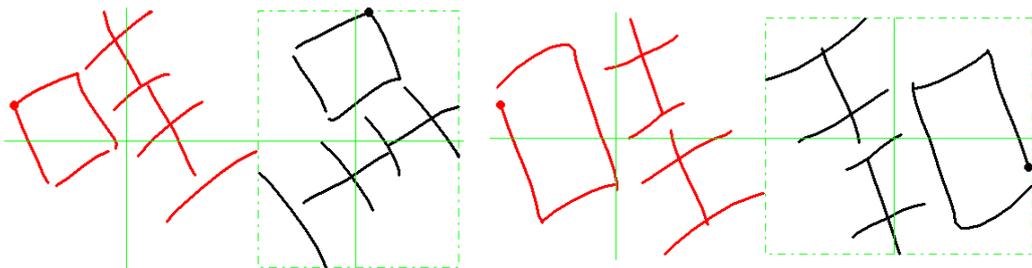


圖 2-2-1 漢字書寫起點示意圖

日文的漢字的書寫方向與中文相同，利用橫筆劃與豎筆劃相互垂直的特性，以及字本身的方向與文字行的方向相互垂直的特性，〔11〕在決定文字行的方向之後，將 X 軸移轉至文字行的方向，Y 軸移轉至其垂直方向，然後求取文字筆劃在移轉後的各個方向的投影量。由結果中觀察發現，在文字行的方向以及其垂直方向的投影量會出現兩個比較集中的峰值，前者較長且為文字的水平筆劃的方向，後者較短且為筆劃的垂直方向，由此即可確定文字的方向。圖 2-2-2 推論求取文字行走向方法示意圖，顯示在文字行之中如何計算推論文字行方向，透過持續觀察前一個筆畫終點與下一個筆畫起點的方向與長度，如果突然轉向近於  $180^\circ$  而且長度遠大於門檻值的話，就可以假設換行事件發生，並且這一行的方向就可以由第一畫起點到其他所有組成筆畫的起點的收斂方向計算出來。其他較複雜的換行偵測方式在此不作贅述。

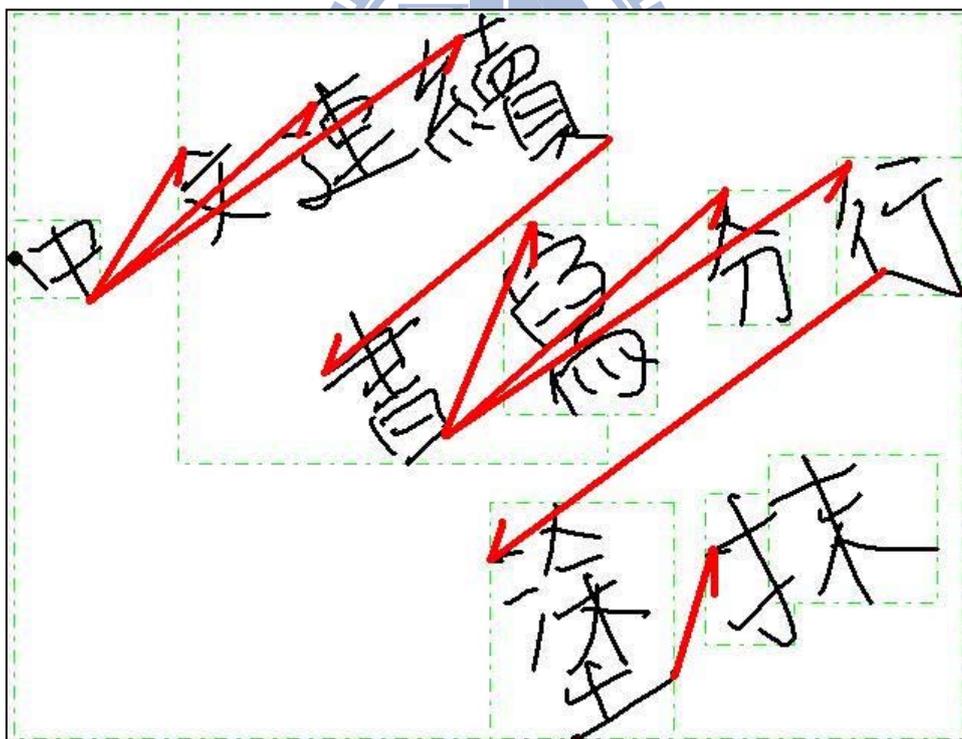


圖 2-2-2. 推論求取文字行走向的方法示意圖

### 第三章 研究方法與系統架構

本節將詳細說明所採用的理論基礎與研究方法、測試資料與評估的依據、以及整個系統的架構，並且針對各個論點提供圖示說明。

#### 3.1 研究方法

觀察漢字楷體的書寫習慣，以及在參考文獻[2]中所參考的漢字筆畫統計，歸納得到兩種漢字的書寫特性：第一種書寫特性是「由左至右，從上到下」，本文中所提到的「水平筆畫」，又稱做「橫筆」，大多都是由左至右的書寫；而文中另外提到的「垂直筆畫」，又稱做「豎筆」，則大多是從上到下的書寫，如圖 3-1-1 漢字水平與垂直筆畫分解圖 所示。

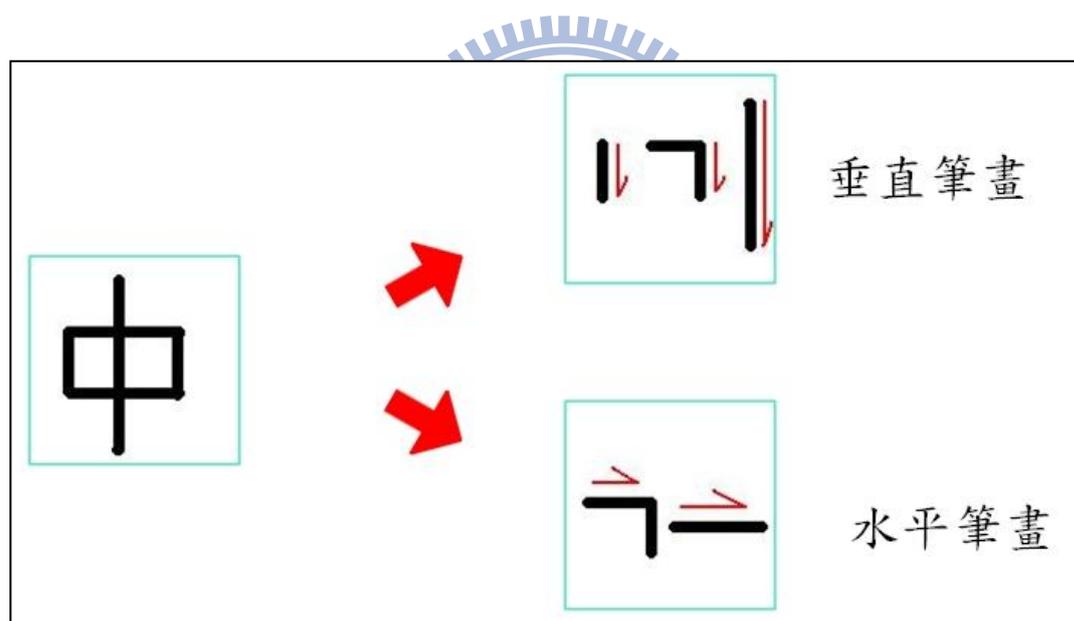


圖 3-1-1. 漢字水平與垂直筆畫分解圖

第二種漢字書寫特性是「方塊字」，在本文中指的是複合筆畫的轉折多半近似直角，少有圓弧。所謂的「複合筆畫」，則是指在一個筆畫中包含多個行筆或者說是書寫方向，圖 3-1-2 複合筆畫例圖舉了三種複合筆畫的例子。

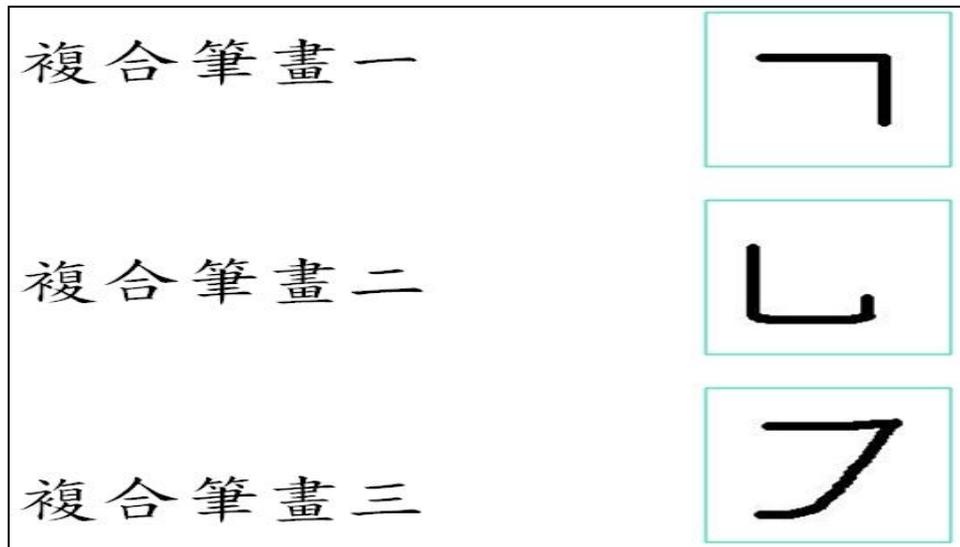


圖 3-1-2. 複合筆畫例圖

楷體書寫的漢字按照其字形中水平筆畫與垂直筆畫的數目以及長短，可以分為下列五種：只有水平筆畫的字、只有垂直筆畫的字、水平與垂直筆畫都沒有的字、水平筆畫與垂直筆畫都有但是都很短的字、以及兩種筆畫都有的字。圖 3-1-3. 以筆畫方向與組成分類的漢字例圖顯示了上述五種分類所指的漢字的部分範例。

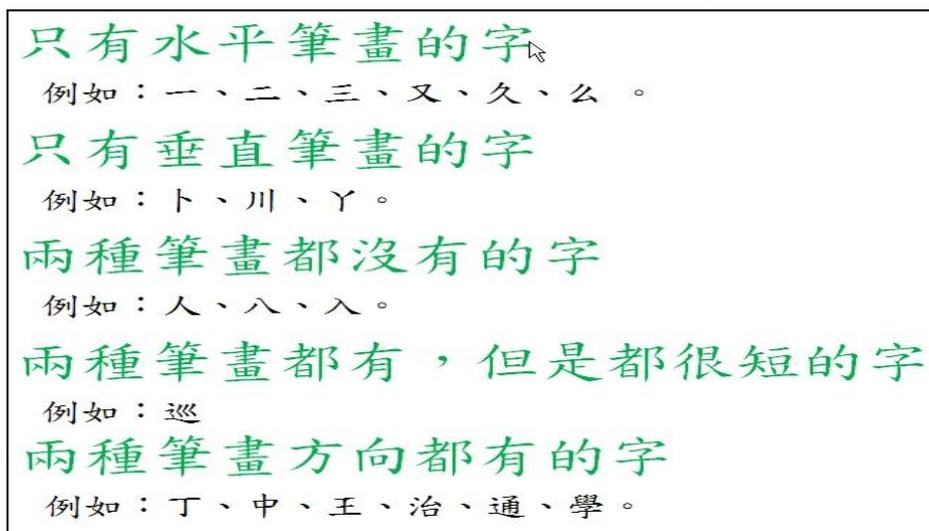


圖 3-1-3. 以筆畫方向與組成分類的漢字例圖

所謂的「筆畫方向」，在本文中所指的是從筆畫的起始點到筆畫的結束點所形成的射線的方向；但是手寫字的筆畫因為書寫習慣或者是美觀的因素，很難形成一條筆直的線，也有如同圖 3-1-2 所提到的複合筆畫，所以又需要引入「線段」，也就是前一個採樣點到下一個採樣點所形成的射線，而這個射線的方向就稱為「線段方向」。在圖 3-1-4(a)(b)(c) 筆畫方向與線段方向示意圖 中，可以得到更為直觀的了解：圖 3-1-4(a) 是原始的筆跡，圖 3-1-4(b) 在原來筆跡上的有箭頭符號的射線就是「筆畫方向」，圖 3-1-4(c) 把原始筆畫簡略成為四個採樣點得到的三個「線段方向」，以箭頭符號在原始筆跡的旁邊標注出來。

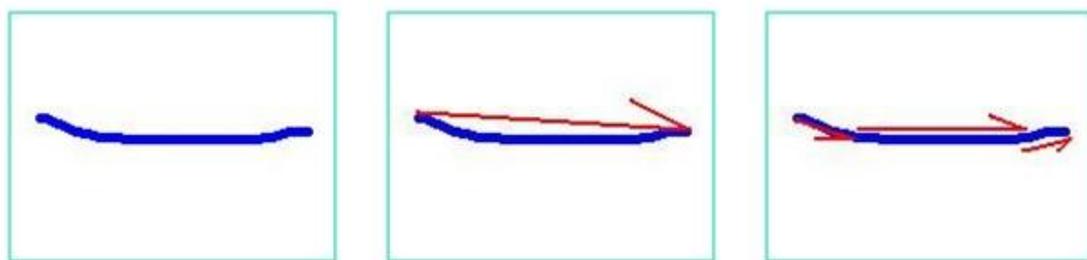


圖 3-1-4(a)(b)(c). 筆畫方向與線段方向示意圖

字的傾斜可以大致分為兩種情況：第一種是字體內部的傾斜 (Internal Slant)，也就是說紅色基線 (Baseline) 維持水平，但是筆畫有一定程度的傾斜，如圖 3-1-5(a)(b) 字體內部傾斜範例所示，圖 3-1-5(a) 表示原始的字，圖 3-1-5(b) 表示有內部傾斜的字，紅色水平線表示基線，基線與字的橫向筆畫平行；紅色垂直線是參考線，表示原本字的縱向筆畫的方向，現在與黑色的縱向筆畫呈現一個夾角  $\theta$ 。

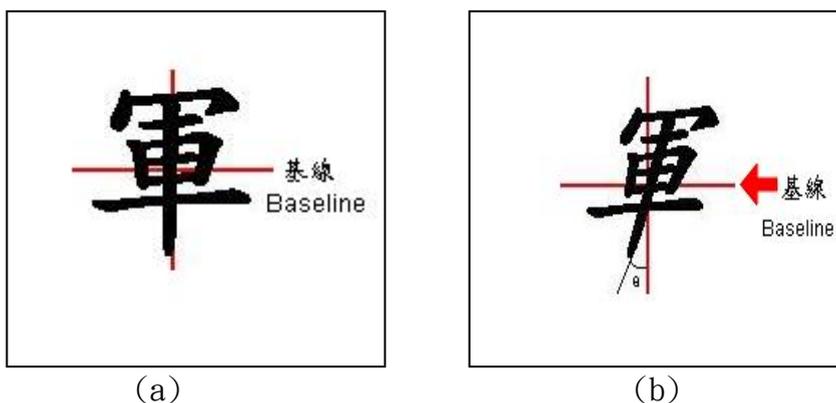


圖 3-1-5(a)(b). 字體內部傾斜範例

第二種是字體外部的傾斜(External Slant)，紅色基線的角度已經改變，與藍色水平軸不再平行，但是字體內部的橫向筆畫與縱向筆畫的夾角還是維持垂直。如圖 3-1-6(a)(b)字體外部傾斜範例所示，圖 3-1-6(a)是原始圖形，圖 3-1-6(b)表示有外部傾斜的範例，與字體的橫向筆畫平行的紅色線段是基線，與字體的縱向筆畫平行的紅色線段則是參考線，這兩者還是維持著互相垂直的關係；但是這兩者與藍色的水平軸以及縱軸都不再是平行的關係。

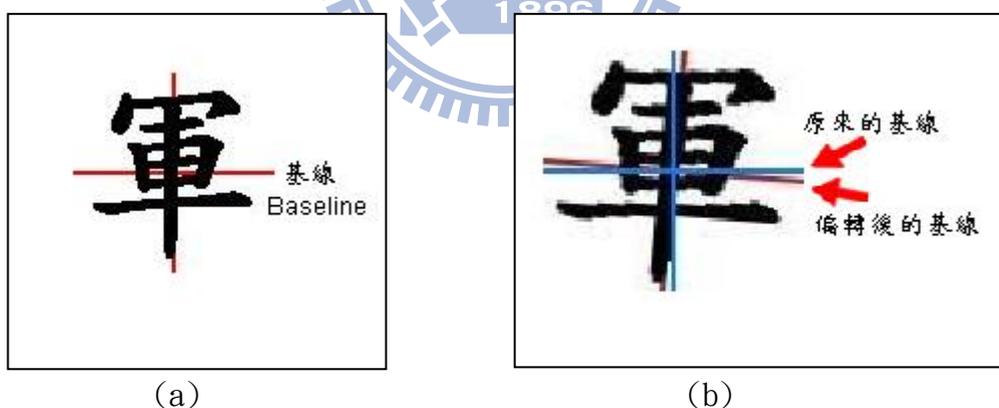


圖 3-1-6(a)(b). 字體外部傾斜範例

本文所指的旋轉是各種角度的字體外傾斜的聯集(Union)，也就是說基線與水平軸的夾角是 $[0^\circ, 360^\circ)$ 中的任意角度。一般說來，手寫字體大多同時包含字體內與字體外這兩種情況的傾斜，如圖 3-1-7(a)(b)(c)(d)同時具有外部與內部傾斜的手寫字範例所示，與字體的橫向筆畫平行的紅色線段是基線，與字體的縱向筆畫平

行的紅色線段則是參考線。字的水平筆畫與基線的夾角為 $\theta$ ，字的縱向筆畫與參考線的夾角則是 $\theta_2$ 。手寫字很容易可以同時找到這兩種傾斜。

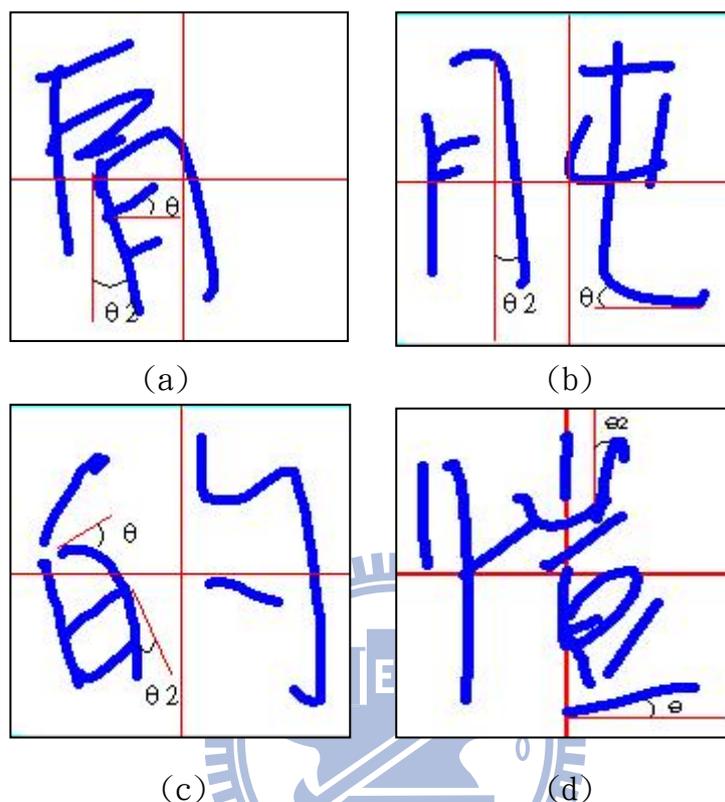


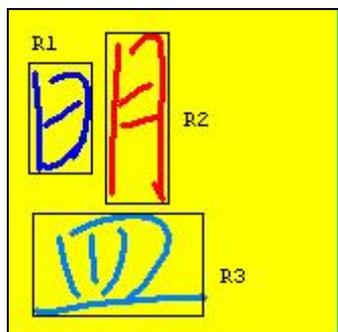
圖 3-1-7(a)(b)(c)(d). 同時具有外部與內部傾斜的手寫字範例

經過上面兩段的說明，我們認為「旋轉」可以說是把「外部傾斜」推到極限，也就是說傾斜的角度很大；而「外部傾斜」則是「旋轉」的特例，也就是旋轉角度很小。本研究所針對要解決的就是「外部傾斜」的問題。

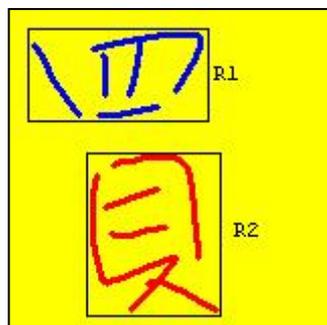
隨著統計分析方法的應用與改善，現在的識別技術可以有效的吸收字體內的傾斜變異；對字體外的傾斜也有一定的吸收能力，以下三段我們將針對這些部分我們會做更深入的說明。

結構分析法的手寫識別核心必須要先把漢字區分為若干的元件 (radical)，圖 3-1-8(a)(b)(c)(d) 漢字元件拆分範例，R1/R2/R3 分別表示被拆分出來的元件，圖 3-1-8 (a)(b)(c)(d) 顯示了元件可

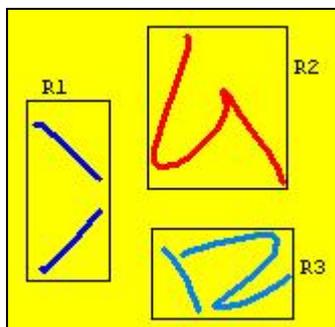
以被組合的位置，其中圖 3-1-8 (c)圖的 R2 與 R3 的劃分其實也是可以合併視為一個元件的，使得分割看起來不會過於瑣碎，不過也失去了原來可以根據筆畫數不同進行彈性組合的特性。



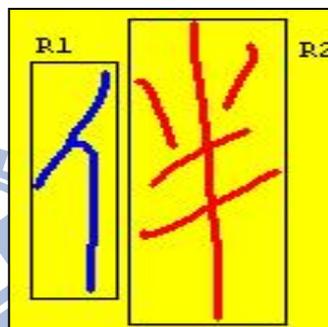
(a)分為上二下一三個元件



(b)分為上下兩個元件



(c)分為左一右二三個元件



(d)分為左右兩個元件

圖 3-1-8(a)(b)(c)(d). 漢字元件拆分範例

利用元件之間的組合，減少所需要儲存的空間，可以只修改元件本身結構而不需要更動整個字，最後還可以一次變更所有具有相同元件的字的結構敘述。按照這種方式，使用結構分析法的手寫識別核心只要增加不同傾斜程度的元件結構敘述，就可以吸收字體內的傾斜變異；這個方法的缺點則是對於字體外部的傾斜變異的吸收能力就比較差了，因為這必須要修改整個字的每一個元件，如果要吸收的外部傾斜角度越多種，就要儲存越多套的結構敘述。而且要使用自動化方式建立這樣的結構描述並不容易，通常必須輔助以大量的人工檢查，並解人工檢查不免流於個人主觀的觀察結果，也就是所謂的經驗法則。因此現在這種方法多半用於前處理或者是後續的細部檢查等輔助識別的用途，或者是筆畫不多且容易書寫的快速手寫輸入識別系統。

統計分析法的手寫識別核心，例如使用 Hidden Markov Model (HMM) 或者是 Neural Network (NN) 演算法的手寫識別核心，則是利用多人收集的手寫軌跡樣本訓練 (training) 後的結果進行識別，樣本的變異越多越齊全，不管是字體內部或者是字體外部的傾斜變異，吸收能力就越強，不過它的上限就是在於所取的特徵對於傾斜旋轉的抵抗能力，越是不受傾斜旋轉影響的特徵，抵抗能力就越強，變異的吸收能力也就越好。由於統計分析法可以使用電腦的自動化處理能力，透過不斷的處理/反饋/調整等三個步驟，求得訓練樣本的最佳化結果，所以是目前最受歡迎的主流識別方法。

我們的目的在加強對字體外傾斜的處理能力，使得原本無法被識別的字符，在消耗少量的計算時間以及儲存空間的前處理之後，可以有機會被辨識核心辨識出來。圖 3-1-9. 本研究系統定位圖 說明了本研究在整個手寫系統中所扮演的角色。

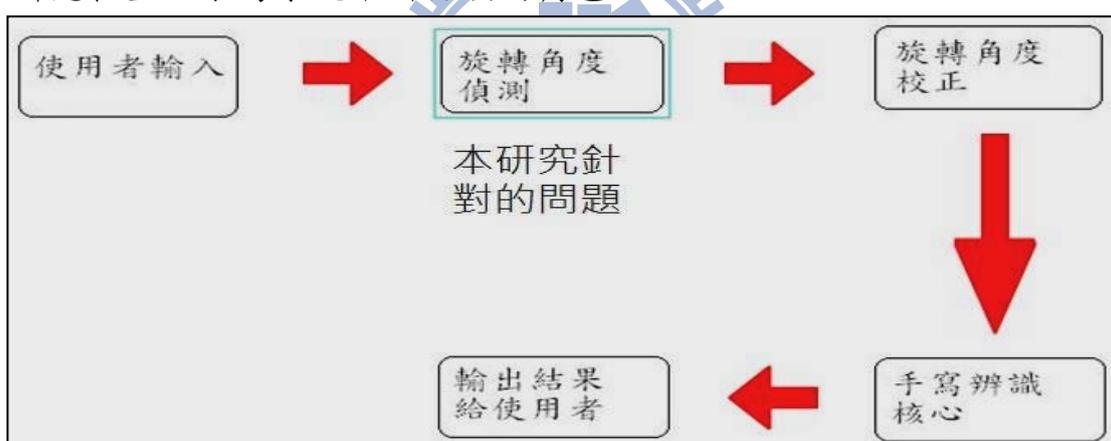


圖 3-1-9. 本研究系統定位圖

根據統計，中文字書寫的時候，出現橫豎筆畫的機會佔大多數約 39.51% 與 33.94% [2]，這兩個筆畫的方向一般是橫筆由左至右，豎筆由上至下，如果一個字裡面同時有出現橫筆與豎筆，那麼我們就可以利用這兩種筆畫現在的方向推測出大致的旋轉角度。圖 3-1-10(a)(b)(c)(d) 橫豎筆畫相對關係示意圖，橫向的筆畫 (X 軸正向的射線) 使用紅色線段標註出來，並且賦與名稱 H；縱向筆畫 (Y 軸負向的射線) 使用綠色線段標註出來，並且賦與名稱 V；H 與 V 的夾

角舉了兩個例子分別是 $\theta_1$ 與 $\theta_2$ 。圖 3-1-10(a)顯示軌跡沒有傾斜的時候的橫豎筆畫夾角，圖 3-1-10(b)~(d)分別表示整個字傾斜甚至旋轉某個角度之後，橫數筆畫的夾角 $\theta_1$ 與 $\theta_2$ ，可以看出 $\theta_1$ 與 $\theta_2$ 在傾斜/旋轉之後，依然還是維持近似直角( $90^\circ$ )；換言之，橫向筆畫與縱向筆畫的關係是「旋轉不變的」。

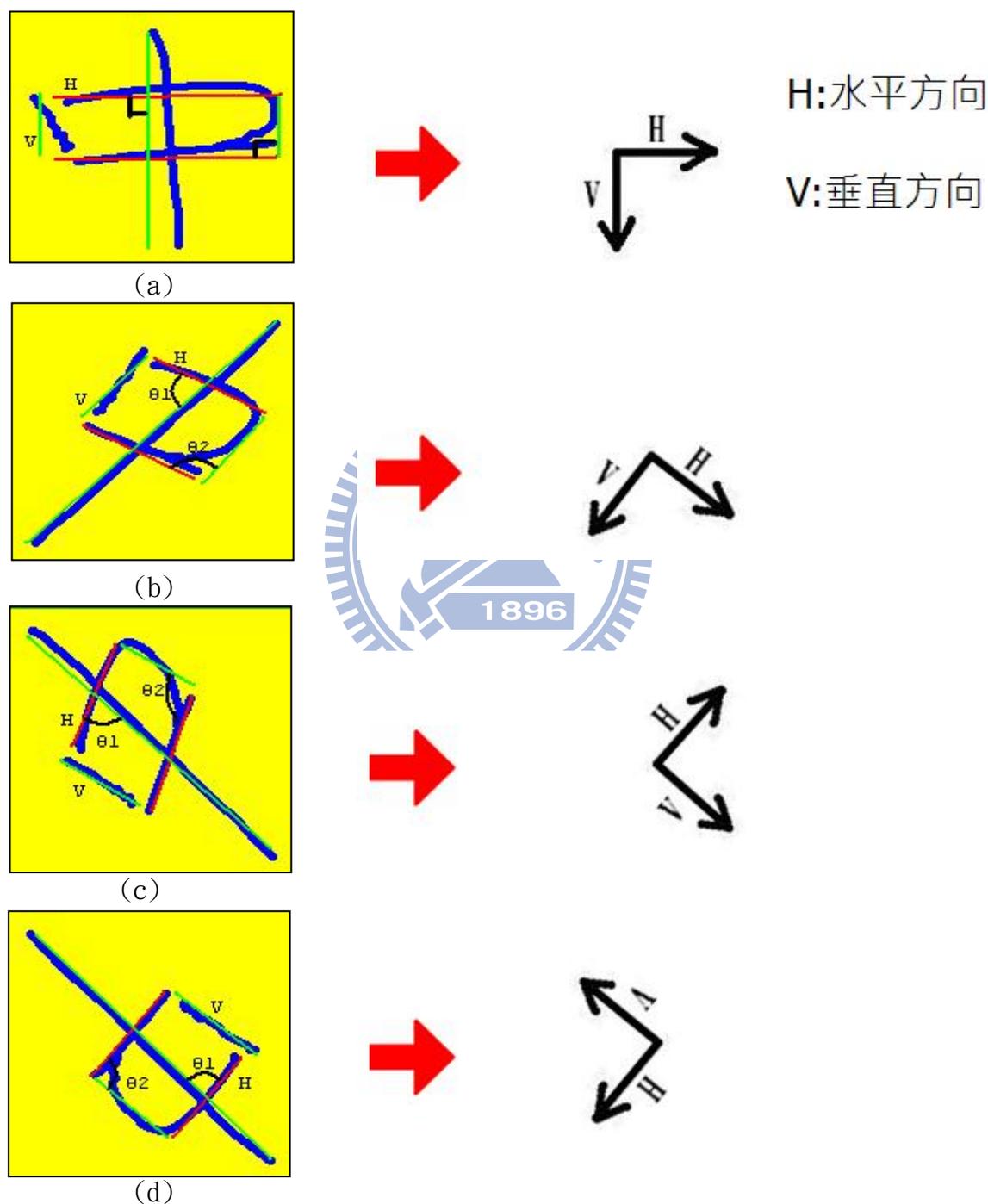


圖 3-1-10. (a)(b)(c)(d)橫豎筆畫相對關係示意圖

對於無法偵測到兩個互相垂直方向的字，或者是可以找到兩個互相垂直方向但是卻是錯的，例如：人、八、入等，本文引入了「經驗法則」作為補救的方法。所謂的「經驗法則」，就是透過觀察這些無法偵測的字，其物理結構的特徵是否足以彌補旋轉偵測方法上的缺陷？如果找得到的話，這些特徵就可以用來確定所要偵測的角度，在此我們舉出幾個字作為例子。圖 3-1-11(a)(b)(c)經驗法則算法的實例運用示意圖 透過「一」、「二」、「三」這三個字來說明與展示找尋物理結構特徵的基本概念以及其產生的演算法。由這三個例子，我們可以看到首先都是要確認筆畫數、筆畫的行筆方向、以及筆畫的長度以免發生算法運用到錯誤的字的情況；如果是多筆畫的字，則更進一步檢查筆畫之間的關係，把這些資訊整合起來，就是形成了「經驗法則」。

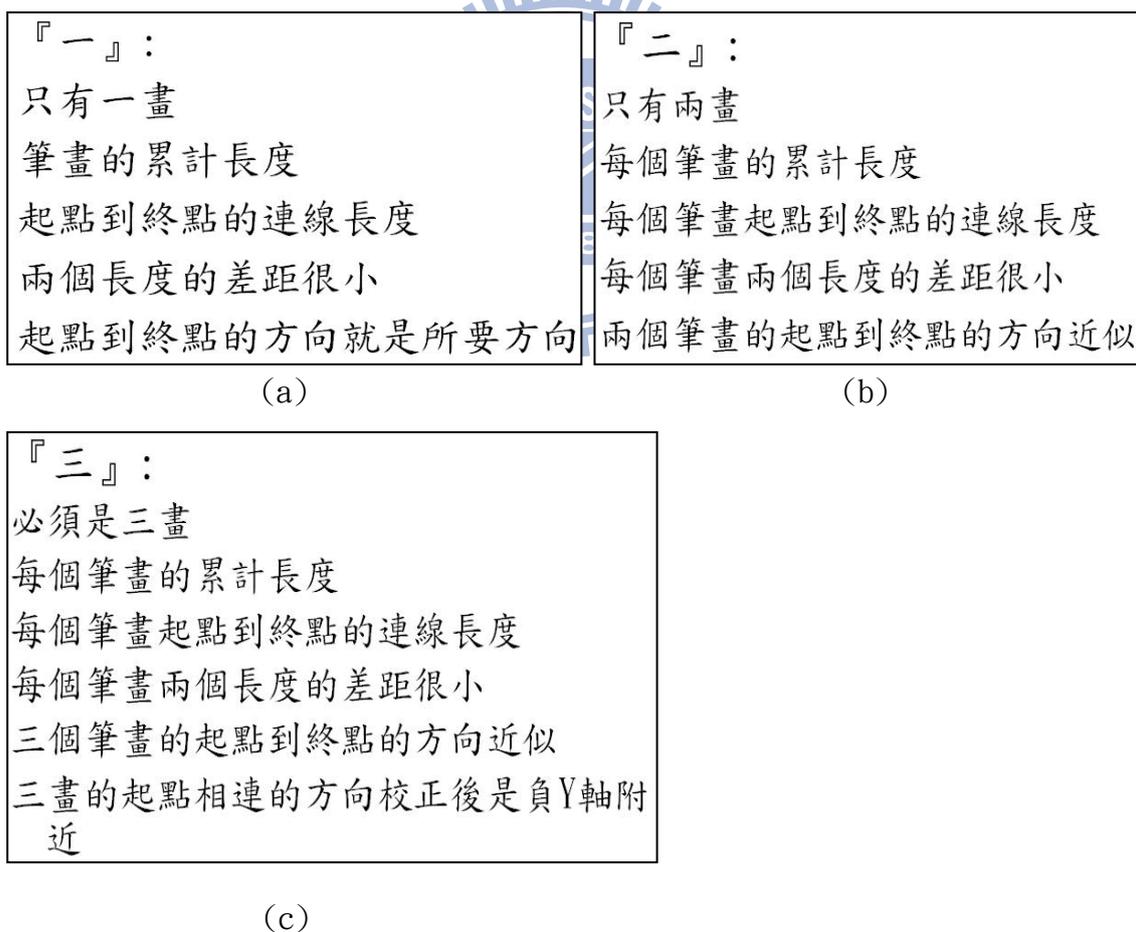


圖 3-1-11(a)(b)(c). 經驗法則算法的實例運用示意圖

### 3.2 系統架構

要偵測一個中文字的旋轉角度，可以分為以下的幾個步驟：(1)筆畫線段的抽取。(2)各個長度方向的累計。(3)尋找累計長度最大的方向。(4)尋找長度足夠的垂直方向。(5)找出滿足中文特性的方向組合。(6)應用經驗法則修正偵測的結果。以下將針對各個步驟做詳細的說明。

對於「筆畫線段的抽取」，必須要考慮三個方向：(1)決定方向的量化方式。(2)抹平，也就是軌跡雜訊的移除。(3)紀錄線段方向與長度。我們首先討論關於決定方向的量化方式。平面座標系統可以被 X 軸與 Y 軸分為四個象限，右上角是第一象限，左上角是第二象限，左下角是第三象限，最後右下角則是第四象限。我們按照象限的編號方式逆時針把整個平面等分為 N 個方向如 圖 3-2-1(a)(b) 方向畫分與量化示意圖 所示。圖 3-2-1(a) 以圓點為中心將圓周 16 等分為例，各個方向是由原點，也就是 X 軸與 Y 軸交叉點向外的射線，四個象限總共可分為 0 到 N-1 的方向編號，圖 3-2-1(b) 是根據方向畫分取 N=16 的時候的量化過程，將每個方向用一個位元(bit)來表示，方向 0 到 15(N-1) 分別對應一個位元組(byte)中的 bit0 到 bit15，如果找出來的線段方向是在方向 0，則對應的 bit0 將會被設置為 1，否則就設置為 0。

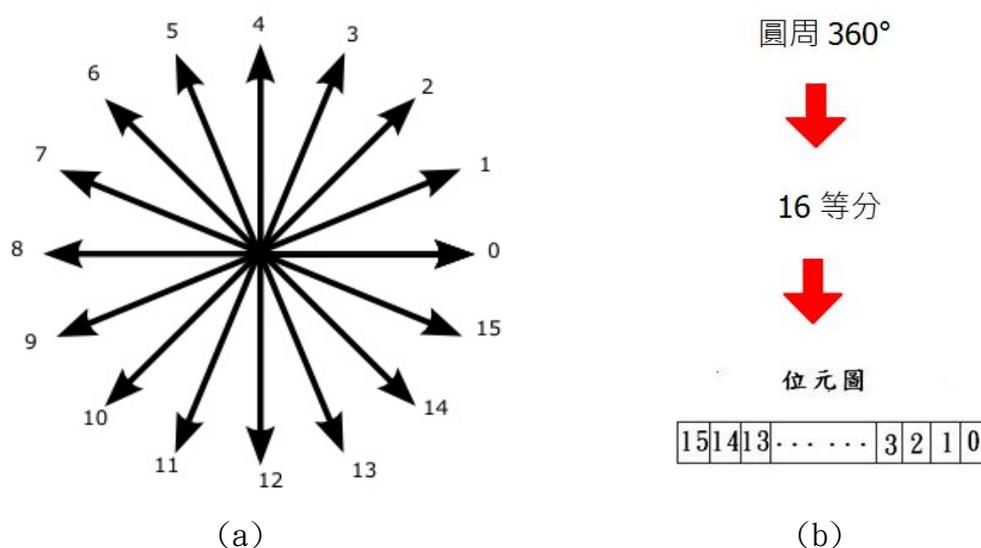


圖 3-2-1(a)(b). 方向畫分與量化示意圖

經過這樣的一個量化(Quantization)的過程之後，我們就可以用整數的位元圖(bitpattern)來表示角度範圍，在運算的表達上也更為方便，如果要表示相鄰的幾個角度的區域的合併，只要簡單的把代表這幾個區域的位元圖加起來，即可以獲得；之後如果要增加合併區域或者刪除兩側的合併區域，則將位元圖之中對應的位元設立(set)或者清除(reset)就可以完成運算。圖 3-2-2 量化方向表達示意圖說明當圓周畫分為 16 等分的時候，要如何表示角度 $\theta$ 落在區間  $[0^\circ, 56.25^\circ]$ ？16 等分的圓周，每一等分相當於  $22.5^\circ$ ， $56.25^\circ$  則剛好落在方向 2 與方向 3 之間，分別以 L0 代表方向 0，L1 代表方向 1，L2 代表方向 2。

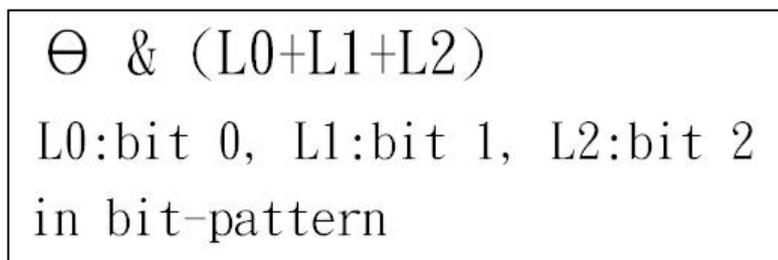


圖 3-2-2. 量化方向表達示意圖

接著我們討論如何移除書寫軌跡中的雜訊，也就是所謂的「抹平」。這個步驟的目的在於降低雜訊干擾的影響，以及減少線段的數目，以下將說明如何達成這兩個目的的方法。透過前一個步驟將所有輸入的線段的方向量化之後，開始依序追蹤每一個筆畫中的線段，每次都是查看兩個線段：追蹤線段與新線段。如果新線段的長度小於門檻值，將會被捨棄，以消除雜訊並減少線段的數目。長度大於門檻值的新線段則考慮其方向與追蹤方向的夾角，如果這個夾角小於門檻值，則兩個線段可以合併為長度更長的追蹤線段，如果夾角大於門檻值，則在記錄原來的追蹤線段的方向與長度後，把新的方向與長度當作新的追蹤線段與方向，繼續整個流程，直到每個筆畫的每個線段都被查看過一次。

針對一個未知的輸入，將所有筆畫拆解成的線段之後，依照量化方向進行長度累計，得知在每個方向上的總長度。第二步驟從累計的各個方向中，找一個長度最大的方向當作起始方向(major direction, M)，再以起始方向為索引，找看看有沒有與它垂直的長線段？如圖 3-2-3 由 major direction M 找尋 second major direction S 所示，黑色實線的射線表示目前所決定的起始方向，major direction M；紅色虛線代表須要找尋的垂直線段，second major direction S。問號表示目前未知是否真正存在，兩個紅色射線 S1 與 S2 表示這兩個方向都可能是我們所要尋找的真正的 S，若都找的到，選取較長的那一個做為返回的結果。

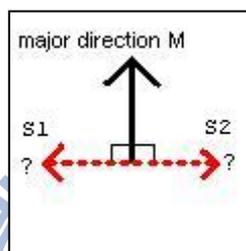


圖 3-2-3. 由 major direction M 找尋 second major direction S

如果找得到這樣的垂直線段，而且是剩餘的所有線段中比較長的，那麼這個就是第二起始方向(second major direction, S)。如果找不到這樣的垂直線段，也就是算法無法自動偵測的情況，必須使用經驗法則加以補充；若經驗法則還是無法適用，那麼就不進行之後的校正。

接著是決定 M 與 S 的夾角，也就是找出滿足中文書寫特性的方向組合測試，以下說明如何完成這個測試並且決定其結果是否正確：這兩個方向是互相垂直的，或者近似於互相垂直，而且橫筆一定指的是 X 軸正向的射線(方向 0)，豎筆則是 Y 軸的負向射線(方向  $3\pi/4$ )，因為我們利用的資訊就是手寫漢字中的橫豎筆劃的資訊。但是哪一個才是 X 軸(水平方向)，哪一個才是 Y 軸(垂直方向)呢？我們先以 M 和 S 中較大者為大數，標記為 `index_large`，較小者為小數，標記為 `index_small`。首先我們試著把大數 `index_large` 假定為方向 0，從

上面的方向示意圖了解到，因為最後要組成一個圓的關係，方向 0 其實也就是方向 N，所以  $factor = N - index\_large$  就是一個補償數值，用來補償  $index\_small$ ，確立它在  $index\_large$  變為方向 0(N) 的之候的修正後結果為  $index\_small + factor$ 。如果修正結果滿足下列條件：它與方向  $3*N/4$  的誤差不大於  $N/8$  的話，就可以確定  $index\_large$  與  $index\_small$  分別是我們所要找的方向 0 與方向  $3*N/4$ 。

另外一種情況就是  $index\_small$  才是我們所要找的方向 0，那麼需要補償的數值就變成  $index\_large$ ，但是此時就不需要先計算  $factor$ ，因為這是一個負向的補償，也就是說  $index\_large - index\_small$  就是補償後的數值；如果這個補償後的數值也滿足第一種情況所提到的條件，就可以確定  $index\_small$  與  $index\_large$  分別就是所要找的方向 0 與方向  $3*N/4$ 。圖 3-2-4. 中文書寫特性測試示意圖 顯示本論文如何進行滿足中文書寫特性的方向組合測試。帶有箭頭方向的虛線表示軌跡校正的前進方向，也就是說把虛線起點的射線的方向當作是筆畫的水平方向的話，另外一個射線將會指向何方，結果就是短而粗的箭頭方向右邊所形成的樣子。如果這樣的組合是正確的，旁邊用圓圈表示；如果是不正確的組合，旁邊用 X 表示。

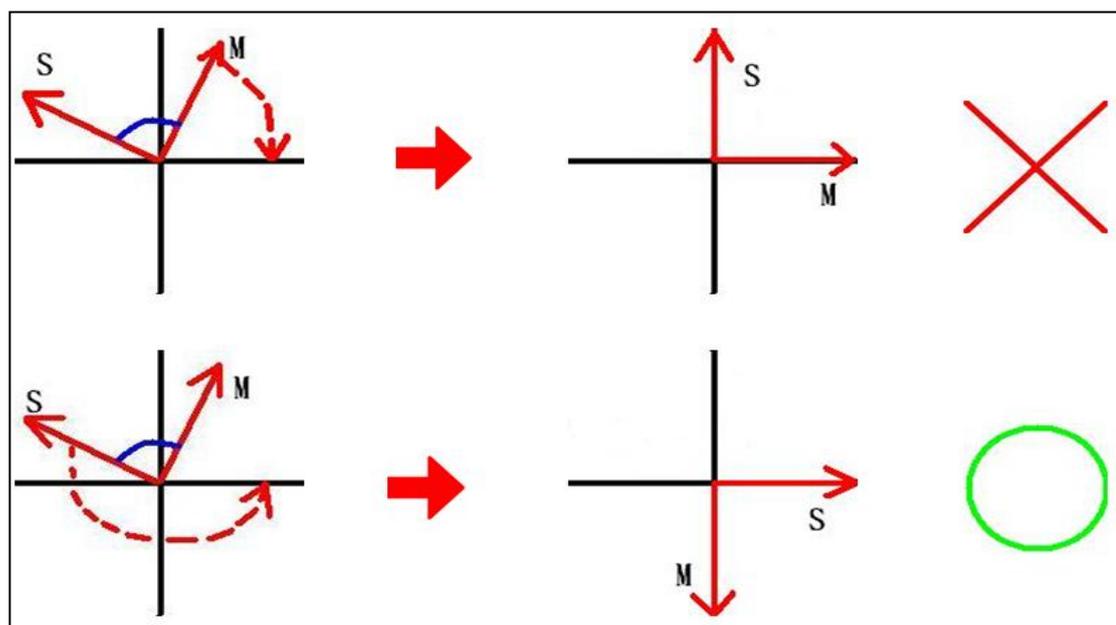


圖 3-2-4. 中文書寫特性測試示意圖

以下是偵測算法，本文稱之為「筆畫線段投票法」的 pseudo-code，返回 0 表示無法偵測或者不需要校正，其他非 0 值表示這個方向的線段必須要轉為 X 軸正向射線的線段，X 軸正向的射線被標記為 L0：

Algorithm 1.: 筆劃線段投票法。

NO\_DIRECTIONS = N(segment clusters 數目的上限)。

remove small segments from ink.

find out ink segments and their length from removed ink.

clustering segments and accumulate length for each cluster.

Find out segment cluster with longest accumulated length and mark it as major direction M.

Find out another segment cluster with enough accumulated length and be perpendicular to M, then marked it as second major direction S.

If S cannot be found, Applying heuristic rules then return 0 as L0.

If  $M > S$  then  $index\_large = M$ ,  $index\_small = S$

Else  $index\_large = S$ ,  $index\_small = M$ .

factor = NO\_DIRECTIONS - index\_large.

If  $(|(factor + index\_small) - 3*N/4| \leq N/8)$

return index\_large.

If  $(|(index\_large - index\_small) - 3*N/4| \leq N/8)$

return index\_small.

return 0.

Done.

## 第四章 實驗與結果討論

本章將分別進行下列的實驗：第一節測試手寫辨識核心對於旋轉/傾斜的容忍能力；第二節應用筆畫線段投票法測試不同精細度的線段量化對於旋轉偵測效果的影響，從中選定一個對於旋轉偵測效果較佳的量化精細度作為結論；第三節 Principal Component Analysis(PCA)的目的在找出輸入資料中具有足夠代表性的特徵，在這個主題之中，我們希望找出主要分布的兩個軸，看看是否就是我們所預期的旋轉傾斜方向，並且與我們所採行的方法進行比較。第四節是基於原來我們所採用的筆畫線段投票法的改善版本的實驗，從中選取表現較佳者建立實驗系統；第五節是兩種旋轉偵測方法的速度比較，第六節則是實驗系統的演示。

在前面曾經提到，手寫識別核心大多具有處理傾斜情況的能力，但是這樣的能力的極限到底在哪裡呢？我們採用在不同時間由不同人所分別收集的繁體中文字常用 5401 字包含其俗體以及簡體寫法的樣本來進行實驗，總計有 7789 種寫法，每一個字的每種寫法有十個樣本，總共有 77890 個原始、未經旋轉的樣本。旋轉樣本的產生則是由原始未經旋轉的樣本，透過程式的座標轉換，產生近似於人工收集的旋轉樣本資料，以每  $28.125^\circ$  為步進值，得到編號 1~12 的 12 組旋轉樣本，作為測試以及評估的基礎。實驗所要量測的數據分別是(1)對於未經過旋轉/傾斜偵測與校正的樣本，手寫識別核心得到可以接受的結果的容忍傾斜角度是多少？(2)對於一個輸入的中文字樣本，偵測的傾斜角度效果最佳的精度大約是多少？(3)由這最佳的精度進行偵測的旋轉/傾斜角度，其校正後的結果對於辨識率的提升有多少的貢獻？(4)進行旋轉/傾斜偵測所需要的時間，占用全部時間的百分比是多少？(5)比較其他方法與我們所使用的方法得到的辨識率提升效果的差距。

#### 4.1 手寫識別核心的傾斜忍受能力

在進行真正的實驗之前，必須要先了解手寫識別核心本身對於傾斜的容忍能力，以免將手寫核心的容錯能力當作是旋轉偵測算法的成果。透過這個前置實驗，我們了解到這個手寫識別核心在傾斜的容錯力是在 $+11.25^\circ \sim -11.25^\circ$  之間，表 4-1-1 顯示了部分實驗的數據，橫軸列編號表示累計前一到十位的候選字識別率；縱軸欄編號表示在多一份角度的差異，步進值為  $2.8125^\circ$ ，逆時針方向傾斜為正，順時針傾斜方向為負。本表表示手寫識別核心在正方向，也就是逆時針方向傾斜的時候手寫識別核心首位與前十位累進識別率的變化，從而得知手寫識別核心對於傾斜角度的容錯能力的極限所在。

橫軸：表示由第 1 位到第 10 位的累進識別率(%)

縱 軸 ： 量 化 值 與 角 度	第1位	第2位	第3位	第4位	第5位	第6位	第7位	第8位	第9位	第10位
0( $0^\circ$ )	97.7%	97.7%	98.9%	98.9%	99.0%	99.0%	99.1%	99.1%	99.1%	99.1%
1( $2.8125^\circ$ )	97.5%	97.5%	98.7%	98.7%	99.0%	99.0%	99.0%	99.0%	99.0%	99.0%
2( $5.625^\circ$ )	96.7%	96.7%	98.3%	98.3%	98.6%	98.6%	98.6%	98.6%	98.6%	98.6%
3( $8.4375^\circ$ )	94.7%	94.7%	96.8%	96.8%	97.3%	97.3%	97.4%	97.4%	97.4%	97.4%
4( $11.25^\circ$ )	90.8%	90.8%	93.7%	93.7%	94.3%	94.3%	94.5%	94.5%	94.6%	94.6%
5( $14.0625^\circ$ )	82.5%	82.5%	86.3%	86.3%	87.5%	87.5%	87.8%	87.8%	87.9%	87.9%
6( $16.875^\circ$ )	67.5%	67.5%	72.2%	72.2%	73.8%	73.8%	74.3%	74.3%	74.6%	74.6%

表 4-1-1. 手寫識別核心對於逆時針方向的傾斜容錯能力調查(部分)

從上表可知，在逆時針方向傾斜角度為  $4(11.25^\circ)$  的時候，首位識別率尚有九成以上，還在可以接受的範圍，之後就急遽的下降到難以忍受的程度。從順時針方向進行的手寫識別核心傾斜容錯能力偵測所得到的結果也是類似的情況。圖 4-1-1「璇」的傾斜角度示意圖顯示了「璇」這個字在  $+16.875^\circ \sim -16.875^\circ$  的若干姿態。圖 4-1-1(a) 是沒有旋轉或者傾斜的樣本，圖 4-1-1(b)~(g) 是分別向右旋轉  $2.8125^\circ \sim 16.875^\circ$  的樣本，圖 4-1-1(h)~(m) 是分別向左旋轉  $2.8125^\circ \sim 16.875^\circ$  的樣本。可以看出有些微的不同，但是其實手寫辨識核心可接受的傾斜角度的範圍並不是很大，這也與我們上一章所討論的結論相符合，手寫辨識核心雖然可以利用收集樣本的多樣性來嘗試吸收解決傾斜的問題，但是為了維持辨識率的穩定性，過大的傾斜角度是不可以放置在訓練數據庫之中的。

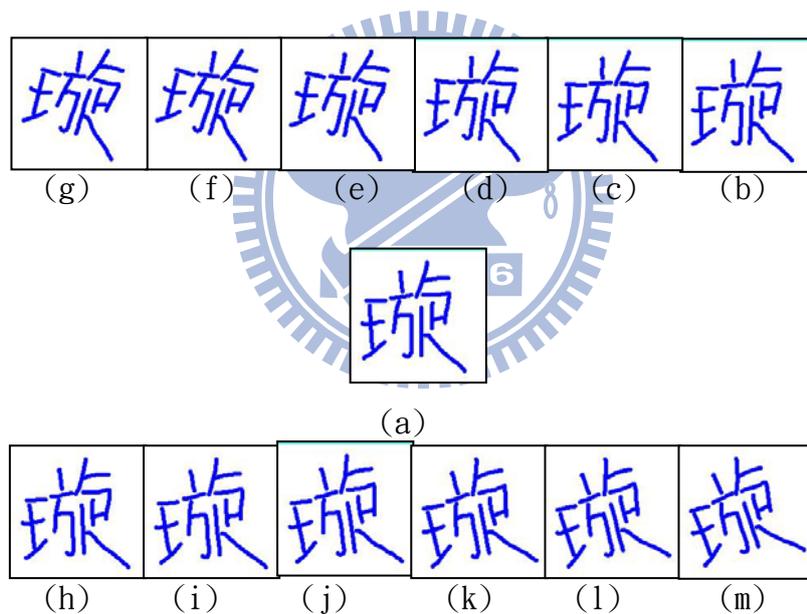


圖 4-1-1. 「璇」的傾斜角度變化示意圖

## 4.2 線段分群的實驗 - 筆畫線段投票法

本節中所要進行的實驗的是筆畫線段方向量化的精細度，也就是不同的量化值對於筆畫線段投票法的影響。因為筆畫線段投票法是針對量化後的方向進行長度的累計，具有較大的累計長度的量化值（方向）就具備較大的影響，這種過程類似把原始資料按照某種觀察或者計算的特徵進行分群（Clustering），所以本文稱之為「線段分群」。

印刷體的識別與書寫體的識別在線段分群的時候可以很直觀的了解到它們的不同：相同字體的印刷體的線段的方向變異通常比較小，而且是比較寬的筆劃再進行細線化之後這樣的差異更被進一步的弭平，因為細線化時通常會選取在筆畫的中央，這樣就不易受到旁邊的雜訊的影響；但是書寫體的線段方向變異就很大了，即使同一個人在前後寫的不同字的的方向的變異也不盡相同，事實上在沒有工具輔助的情況下，即使是工整書寫的手寫字也不容易找到兩個完全平行的筆畫。在這種情況下，線段或者筆畫的分群工作顯得相當不易，不管是傾斜校正或者是現在所做的旋轉校正，能夠正確的分群，是我們所要面對的第一個重大的問題。

首先我們想到的是每一度一等分，總共分為三百六十份，但是從上一節我們知道，核心本身就有一定的容錯力，只要手寫的樣本傾斜角度的偵測與校正後能夠落在這個容錯範圍之內，大部分的字都可以正確識別，所以偵測角度的結果可以不需要十分精確；另外一個不採用這麼詳細的角度區分的原因則是方向太過於分散，每個線段的長度也都很小，反而統計不出來正確的群聚。第三個原因是使用者書寫的筆畫的角度不會是一個絕對穩定的角度，而是一個相對穩定的角度，造成這種誤差的原因可能是手的抖動、書寫習慣的差異、甚至是為了書法美觀而故意造成的，既然是相對的穩定，那麼就必定會有一個判斷的區間，這個區間就是我們進行分群的基礎。

既然核心忍受的傾斜角度大約是  $11.25^\circ$ ，若是把圓周分為 32 等分，應該就可以得到不錯的結果，表 4-2-1 實驗的結果顯示這樣的

預期是正確的，橫軸表示旋轉角度的區分編號以及實際對應的角度；縱軸表示辨識結果的候選字首選累計識別率(1)/(2)，其中(1)表示校正後的識別率，(2)表示校正之前的識別率。由本表可以看到在沒有附加其他限制條件的情況下，32 等分的旋轉角度偵測已經有不錯的結果。

辨識率 \ 角度	20(56.25°)	30(84.375°)	50(140.625°)	60(168.75°)
候選字首選 識別率	63% / 10%	65% / 0%	64% / 0%	65% / 0%

表 4-2-1. 32 等分的旋轉角度偵測後的校正結果識別率

但是如果分為更多的等分，效果是否會變得更好呢？若是會變得更好，那麼何時會反轉呢？我們繼續實驗 2 倍數的等分，64 等分與 128 等分的實驗數據分別參見表 4-2-2 以及表 4-2-3，橫軸表示旋轉角度的區分編號以及實際對應的角度；縱軸表示辨識結果的候選字首選累計識別率(1)/(2)，其中(1)表示校正後的識別率，(2)表示未校正的識別率。

辨識率 \ 角度	20(56.25°)	30(84.375°)	50(140.625°)	60(168.75°)
候選字首選 識別率	62% / 10%	72% / 0%	65% / 0%	63% / 0%

表 4-2-2. 64 等分的旋轉角度偵測後的校正結果識別率

辨識率 \ 角度	20(56.25°)	30(84.375°)	50(140.625°)	60(168.75°)
候選字首選 識別率	63% / 10%	64% / 0%	62% / 0%	65% / 0%

表 4-2-3. 128 等分的旋轉角度偵測後的校正結果識別率

對於直線的雜訊消除則是另一個需要考量的因素，基於種種的原因，例如電子雜訊，人手的抖動，個人的書寫習慣等等，一個直線的筆畫用手寫的時候通常沒有辦法如同我們想像中的那麼完美，會有一個上下左右飄移的歪斜現象，雖然每個筆劃的量或許不是很多，但是眾多筆畫累積起來之後，也會對方向的統計有不小的影響。所以，我們需要把一個線段「抹平」的步驟，換句話說，是要把不規則跳動的點略過，讓方向差異小的線段合併，使得筆畫書寫時候的走向可以更清楚的呈現出來，如圖 4-2-1 軌跡移除雜訊示意圖所示，圖的左邊是原始的軌跡，圖的右邊是經過移除雜訊(抹平)處理後的軌跡，圖中用藍色圓圈標示的部份是平整差異比明顯的地方，數字表示對照的位置的編號。在這些標註出來的地方，消除雜訊後曲折的軌跡變得平直，轉折的部分變的銳利，使得筆畫的方向更清楚地呈現出來。

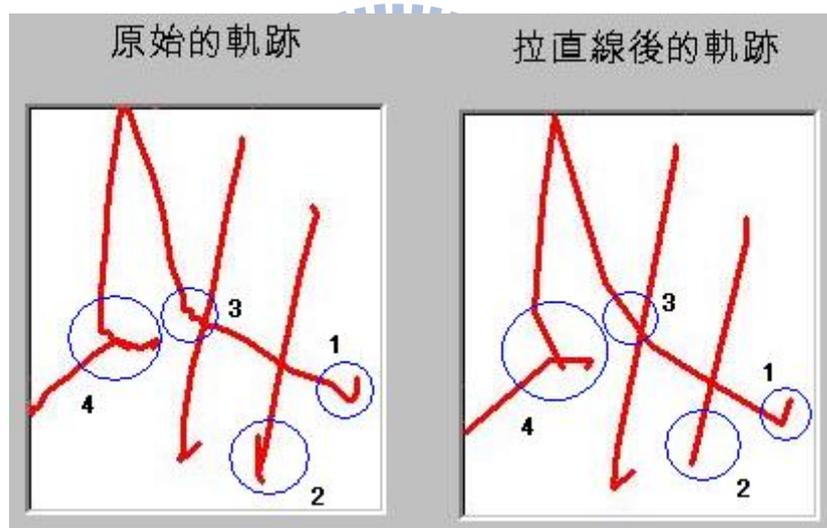


圖 4-2-1. 軌跡移除雜訊示意圖

表 4-2-4 比較未抹平與抹平後的軌跡對平均識別率的影響，顯示沒有抹平與抹平之後偵測結果的差異，橫軸表示累計前十位的識別率，縱軸表示沒有使用抹平函數以及使用抹平函數的累計識別率。從表中的比較可以得知，在移除雜訊(抹平)之後，平均識別率有 4.3%~4.5% 的提升，顯然移除雜訊對於提升偵測效果是有正面意義的。測試的數據是根據本章

橫軸：前 10 位的累計辨識率 (%)

辨 識 率 軌跡源	累進									
	1	2	3	4	5	6	7	8	9	10
原始	53.0 %	53.0 %	55.2 %	55.2 %	56.1 %	56.1 %	56.4 %	56.4 %	56.6 %	56.6 %
抹平	57.3 %	57.3 %	59.7 %	59.7 %	60.6 %	60.6 %	60.9 %	60.9 %	61.1 %	61.1 %

表 4-2-4. 比較未抹平與抹平後的軌跡對平均識別率的影響

### 4.3 使用不同方法偵測旋轉角度-PCA

Principal Components Analysis(PCA)[16] 是一個受到廣泛應用的統計技術，例如人臉的識別、影像壓縮等等，這個方法也被運用在多維度的樣板偵測。透過 PCA 可以找出包含最多輸入資訊的方向，在二度空間上找到的就是兩個相互垂直的軸，對於輸入的文字影像來說，我們希望這兩個軸就代表著這個字的水平方向與垂直方向。使用 PCA 的方法是以輸入的點座標(X, Y)數對為基礎，先求取所有數對的二維 Coveriance Matrix，如 Eq. 4-3-1。為什麼要求取 Coveriance Matrix？這是因為目的是要找出書寫的軌跡分布的兩個軸的關係。沒有旋轉的軌跡當然是存在一般的(X, Y)平面，但是因為書寫習慣的關係，軌跡的分佈主要應該大多是分別沿著 X 軸與 Y 軸的方向，也就是

本文假設的水平筆畫與垂直筆畫，所以我們很直覺的認為如果考慮求取一個輸入軌跡的座標分佈情況，只要找到相對應的兩軸，應該就有機會可以反推回正確的旋轉角度。Coveriance Matrix 的四個元素計算出來之後，就可以利用這四個元素計算出標準差(Standard Deviation)，也就是兩個特徵值(eigenvalue)，再利用這兩個值來求取對應的兩個特徵向量(eigenvector)，作為我們所要處理的主要方向軸(M，水平軸)與次要方向軸(S，垂直軸)。

$$C(X, Y) = (\sum(X_i - X_{avg})(Y_i - Y_{avg})) / (n - 1) \quad i = 1 \dots n \quad (\text{Eq. 4-3-1})$$

以下列出使用 PCA 方法進行旋轉角度偵測的虛擬程式碼。

Algorithm 2.: PCA 角度偵測法。

step 1. Get Coveriance Matrix using input ink data

$$\text{Coveriance Matrix} = \begin{bmatrix} c11 & c12 \\ c21 & c22 \end{bmatrix}$$

step 2. compute eigenvalue 1 and eigenvalue 2

via the Coveriance Matrix

$$\text{eigenvalue1} = (c11+c22)/2 + \sqrt{((c11+c22)*(c11+c22))/4 + c12*c21 - c11*c22);}$$

$$\text{eigenvalue2} = (c11+c22)/2 - \sqrt{((c11+c22)*(c11+c22))/4 + c12*c21 - c11*c22);}$$

step 3. Using LU decomposition and Gauss-Jordan elimination to compute eigenvector 1.

Using eigenvalue 1 and coveriance matrix to build pMatrix.

$$\text{pMatrix} = \begin{bmatrix} c11-\text{eigenvalue1} & c12 \\ c21 & c22-\text{eigenvalue1} \end{bmatrix}$$

$$\text{pMatrix}[0]=c11-\text{eigenvalue1};$$

```

pMatrix[1]=c12;
pMatrix[2]=c21;
pMatrix[3]=c22-eigenvalue1;
step 5. x of eigenvector 1 = decomposed matrix element 1
y of eigenvector 1 = decomposed matrix element 2
value = square root of
      (square(pMatrix[0])+square(pMatrix[1]));
      [value2  value1]
      [0      0]
x = value1 / value;
y = value2 / value;
step 6. return to step 3., using eigenvalue 2 to
compute eigenvector 2
step 7. Compare these two eigenvalues, pick the eigenvector
whose associated eigenvalue is larger as the vertical
axis, and the other is the horizontal axis.
The horizontal axis is what we expect.
Step 8. Done.

```

一般線上手寫識別輸入的都是向量式的點座標，而不是一個二維的影像。向量式的點座標由於只有標記筆畫採樣的點，沒有軌跡經過或者是非採樣點都不計入，所以其儲存空間可以獲得更大的利用率。

首先我們就針對點座標的陣列進行 PCA，圖 4-3-1(a)(b)(c)(d) 沒有旋轉的樣本經過 PCA 運算的部分結果。每一組圖片由兩張圖組成，最左邊的高低線圖表示這個字在這個角度的全部樣本經過 PCA 運算偵測到的角度包含的樣本數目，圖左側是偵測到的垂直軸角度，圖右側是偵測到的水平軸角度，斜線左邊的數字是方向，斜線右邊的數字是偵測到的樣本數；由於我們將一個圓分為 128 方向，所以左側的理想值應該是 32 或者是 96 以及其附近的值，右側的理想

值應該是 0 或 64 或 127 或者是其附近的值。由這些數值可以看出利用向量式的點座標陣列來進行 PCA，它的效果並不是十分的穩定，統計分布的結果與我們所預期的並不相符。在同一組圖片中統計高低線圖的右側的圖片是每一個字的兩個測試樣本，樣本中左側的圖片是用來進行偵測的原始圖，上面綠色的線段是我們預期垂直方向的軸，黑色的線段則是預期水平方向的軸；每個軸的一端有一個小圓點，那是我們預計的起點，也就是一開始計算出來的 eigenvector 的終點；另一端則是 eigenvector 乘以-1 之後的位置。

對照原始圖上面的 eigenvector 的走向，我們無法找到一個相對穩定的規則來推論經過 PCA 運算之後所偵測到的旋轉角度。輸入樣本的差異有時候會讓兩個我們認為相當近似的書寫軌跡，得到與預期相差很大的旋轉角度偵測結果。例如圖 4-3-1(a) 宙與圖 4-3-1(b) 尖，他們的差異看起來並不是很大，有點雜訊而且筆劃傾斜的角度稍為有所不同，但是兩個偵測出來的角度校正之後，很明顯可以看到兩者相差 180°。再由圖 4-3-1(c) 巡看起來，同樣的問題看起來更加清晰，甚至兩個軸完全互換過來，這很顯然的與我們所預期的完全不同。

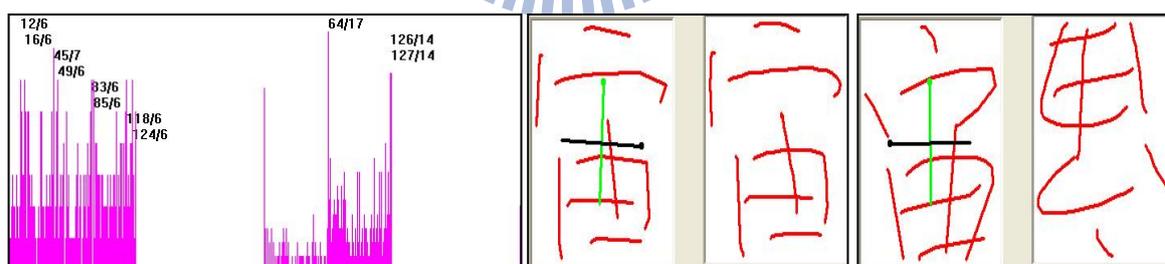


圖 4-3-1(a)

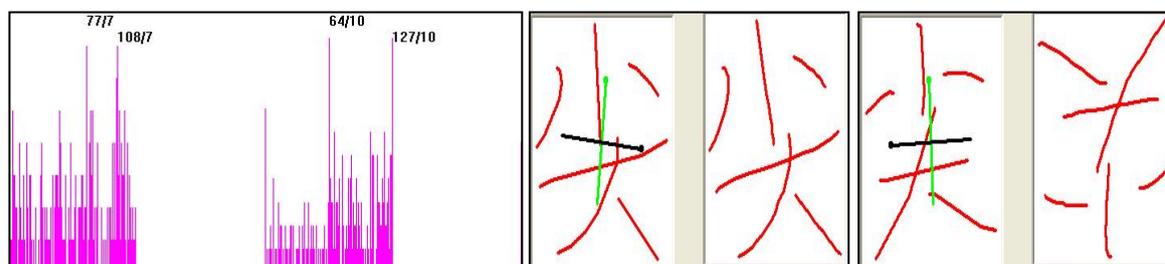


圖 4-3-1(b)

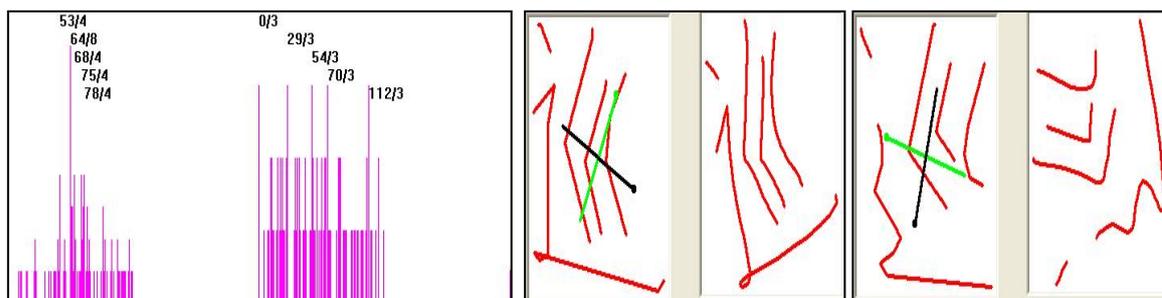


圖 4-3-1(c)

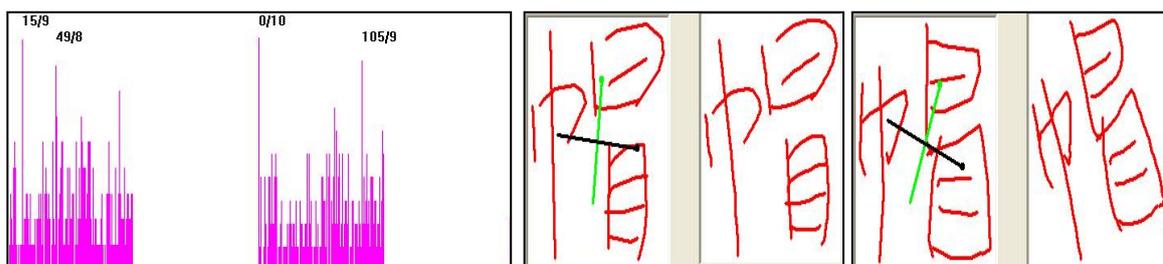


圖 4-3-1(d)

圖 4-3-1(a)(b)(c)(d). 沒有旋轉的樣本經過 PCA 運算的部分結果



至於已經旋轉過後的樣本，經過 PCA 處理之後，兩個特徵向量變化的程度更加難以掌握，圖 4-3-2(a)(b)(c)(d) 旋轉  $112.5^\circ$  後的樣本經過 PCA 運算後的部分結果。比較圖 4-3-1 與圖 4-3-2，以及 PCA 工作的原理，我們可以觀察到 PCA 的結果很容易受到冗餘與錯誤的資訊的影響，甚至只要筆畫傾斜的角度有所差異，即使是在同一個字的相近兩個樣本，也很有可能得到相差很大的結果。所以如果使用 PCA 運算，使用手寫的輸入點座標產生的影像來偵測旋轉的角度，無疑是相當不穩定的。

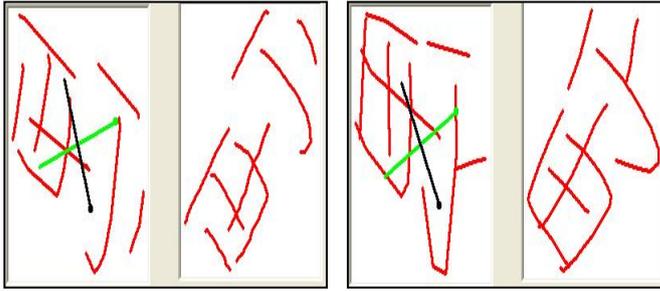


圖 4-3-2(a)

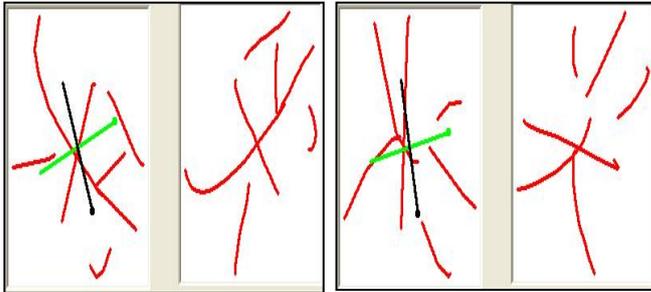


圖 4-3-2(b)

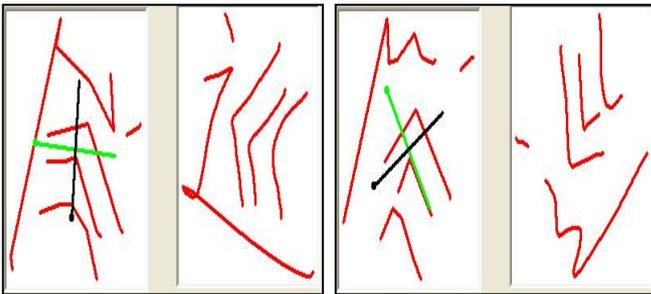


圖 4-3-2(c)

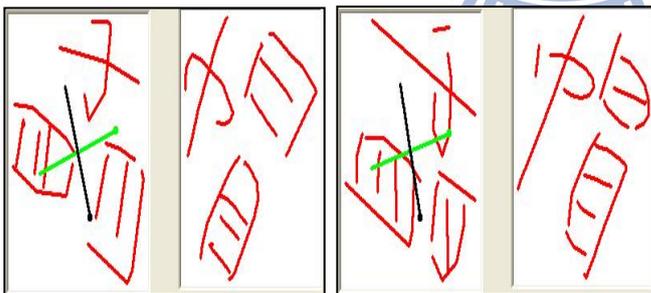


圖 4-3-2(d)

圖 4-3-2(a)(b)(c)(d). 旋轉  $112.5^\circ$  後的樣本經過 PCA 運算後的部分結果

PCA 運算的結果雖然需要加強其穩定性，但是我們預期點座標主要分佈的方向將代表原先 X 軸(水平)與 Y 軸(垂直)方向，還是相當正確的，圖 4-3-3(a)(b)(c)(d) PCA 方法成功偵測旋轉角度案例圖顯示旋轉  $112.5^\circ$  後的樣本經過 PCA 運算後偵測成功的部分結果。

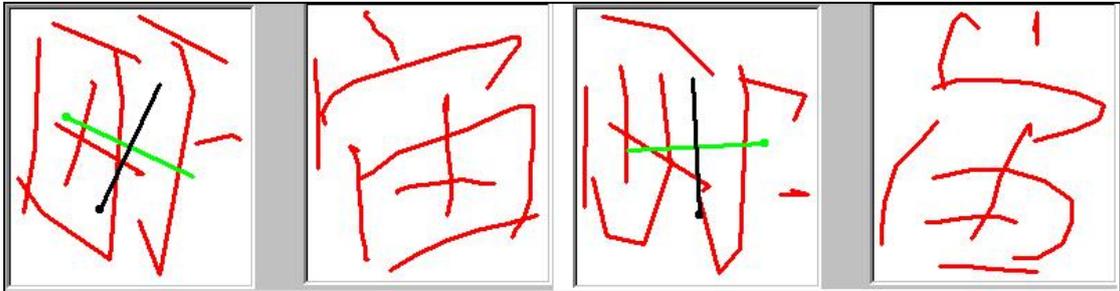


圖 4-3-3(a)

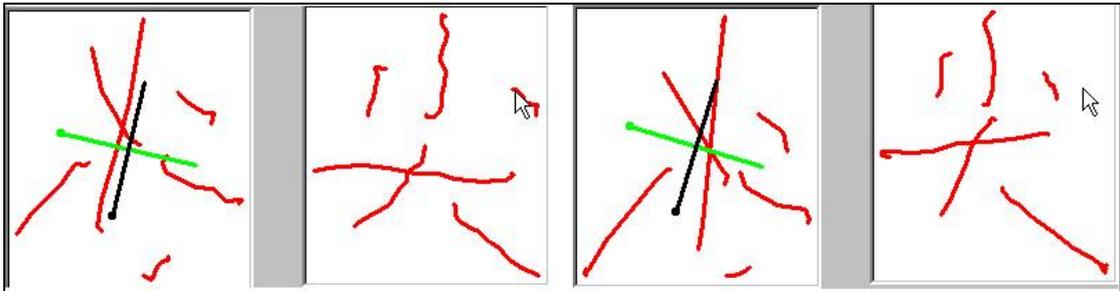


圖 4-3-3(b)

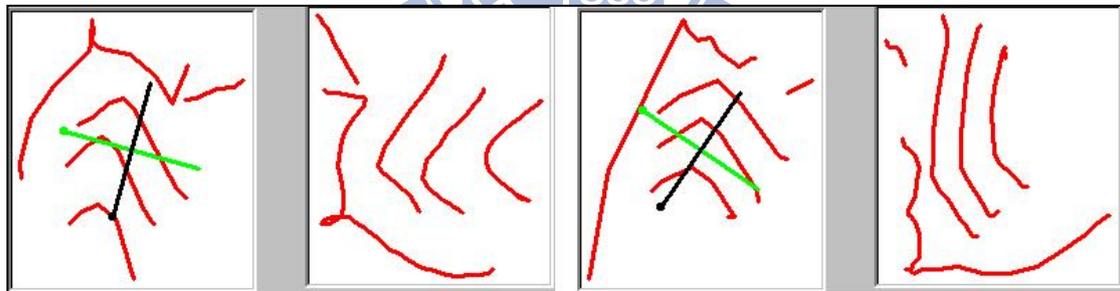


圖 4-3-3(c)

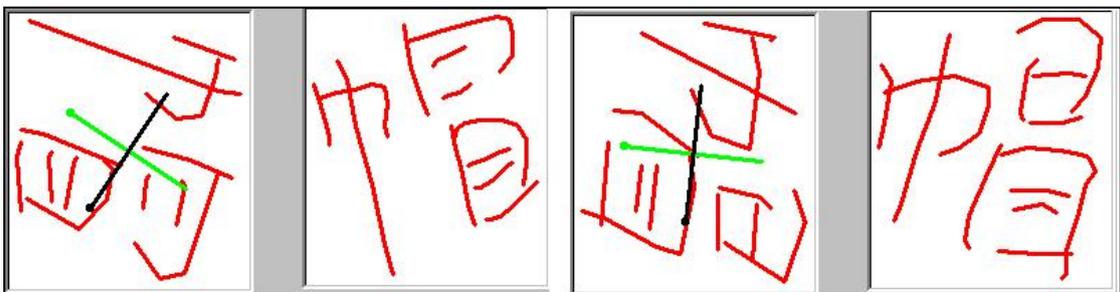


圖 4-3-3(d)

圖 4-3-3(a)(b)(c)(d). PCA 方法成功偵測旋轉角度案例圖

#### 4.4 改善線段分群的表現的實驗

由 4.2 節的實驗數據，我們可以看出對於旋轉的偵測的效果並不是很好，因為對於辨識率的提升的幅度遠不如我們的預期，雖然在特定的角度可以回復到七成的辨識率，但是大多數只有回復到六成五的辨識率；與 4.3 節的 PCA 成果比較，其實兩者的差異不是很大。造成筆畫線段投票法表現不彰的原因，我們分析之後得到的結論是：因為筆劃方向上的微小差異太多，超過我們所預期設置的門檻值，造成這些線段無法與主要方向結合，這就使得我們的假設很難得已成立，我們原先的假設是水平向右的筆劃與垂直向下的筆劃在大多數的情況下應該佔據累積長度的前兩位。無法結合的小線段太多，造成算法找到的互相垂直線段可能與原先的預期有相當大的誤差，如圖 4-4-1(a)(b)(c)(d). 有許多無法合併的小線段的手寫字 所示，這些字的特性是短的筆畫很多，軌跡的平滑性不佳，雜訊以及筆畫的方向變化較大，所以產生出很多的小線段，而且這些小線段的方向都相近於真正的水平線段方向，但是因為方向的差距剛好大於門檻值而無法被合併。

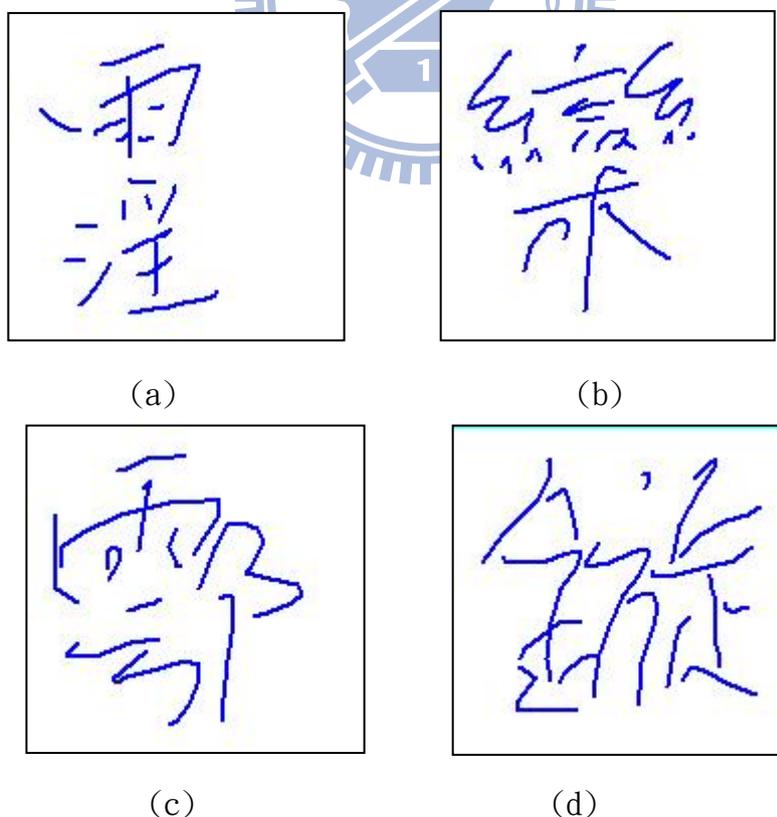


圖 4-4-1(a)(b)(c)(d) 有許多無法合併的小線段的手寫字

為了解決這種的問題，我們發現[2]之中提到的 K-mean clustering 是一個有可能符合我們需要的方法。K-mean clustering 的目的在於找出一群輸入特徵之中的群聚中心，這個方法所以稱為 K-mean，就是它可以找出 K 個群聚的平均中心點。而我們最後的目的就是要找出輸入的筆劃線段中水平向右線段的群聚以及垂直向下線段的群聚。以下將稍微解釋一下我們如何應用這個方法來找出群聚，之後再顯示實驗所得到的數據，證明這個方法真的解決了我們所面臨的困難。

一般的 K-mean clustering 所經常遭遇的幾個問題，首先就是如何決定群聚的數目 K，對於本文所要解決的問題來說，這個 K 是很好決定的，無疑就是 2。第二個步驟是如何決定初始化的中心點，對於特徵資訊的最後形式沒有任何參考依據的情況，初始中心點選取不好，會使得這個方法得到的結果陷入鞍點，也就是區域最佳解 (local-minimum)；但是這樣的情況對於本文所要處理的主題來說，碰到的機會是比較小的，因為前面 4.2 節中已經有提到我們所要找的中心點的樣式：一個最長的線段以及另一個足夠長而且與前一個線段近似垂直的線段，雖然無法保證這樣絕對不會陷入鞍點，但是找到近似最佳解 (approximate global-minimum) 的機會已經增加不少。第三個步驟則是將輸入的點座標數對發布到中心點的群聚之中，計算出新的中心點，並且找出這組解的誤差值，也就是評估前後兩個中心點的誤差是否足夠小？第三步驟的兩個重要因素影響到這個方法的成敗：如何計算出新的中心點以及誤差評估函數的計算方法。對本文而言，後者是相當直觀的，兩次的兩個中心點如果距離都小於門檻值，那麼就可以預期是找到所需要的結果了。重點在於中心點的重新計算，如果採用單純的方向而不考慮長度的話，最後計算出來的結果與筆畫線段投票法的效果相差不大，但是所花費的時間卻是筆畫線段投票法的十幾倍，這很明顯是不可使用的；如果同時也考慮線段的長度的話，那麼這個方法的穩定性以及收斂的速度都非常的令人驚喜：新的中心點的計算方式是將 N 個線段組成的群聚中的每一個線段 S 乘以對應的長度 L 加總之後除以群聚的線段總長度 TL，得到新的中心點

C，也就是這個群聚的新的方向值。公式如下(Eq. 4-4-1, Eq. 4-4-2)：

$$TL = (\sum Li), i=1 \sim N \quad (\text{Eq. 4-4-1})$$

$$C = (\sum Si \times Li) \div TL, i=1 \sim N \quad (\text{Eq. 4-4-2})$$

重複運用第三個步驟一直到新舊兩個中心點的誤差小於門檻值或者是已經套用第三步驟達次數迴圈上限，就以新產生的兩個中心點作為偵測到的主要方向與次要方向。利用上述的 K-mean clustering 方法以及 4.2 節的線段分群-使用筆畫線段投票法的結合，我們得到整個旋轉角度偵測系統對於辨識率的貢獻結果如表 4-4-1 增加 K-mean clustering 後對偵測旋轉角度後的前十位累計辨識率所示。

辨 識 角 度	累 進 率	1	2	3	4	5	6	7	8	9	10
		0(0°)	89.10%	89.10%	90.90%	90.90%	91.40%	91.40%	91.60%	91.60%	91.80%
30(84.384°)		77.30%	77.30%	80.00%	80.00%	80.80%	80.80%	81.30%	81.30%	81.50%	81.50%
60(168.75°)		77.50%	77.50%	80.20%	80.20%	81.00%	81.00%	81.40%	81.40%	81.70%	81.70%
90(253.125°)		73.10%	73.10%	75.80%	75.80%	76.60%	76.60%	77.00%	77.00%	77.20%	77.20%
120(337.5°)		78.50%	78.50%	81.50%	81.50%	82.40%	82.40%	82.90%	82.90%	83.20%	83.20%

表 4-4-1. 增加 K-mean clustering 後對偵測旋轉角度後的前十位累計辨識率

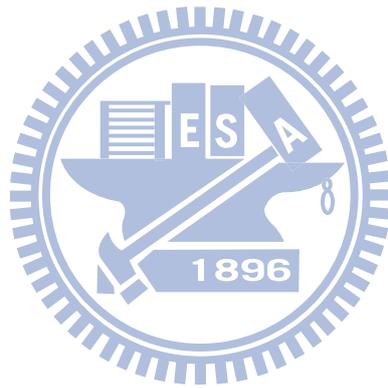
#### 4.5 兩種線段分群方法處理速度比較

我們的希望找到一個不需要占用太多時間或者是需要訓練的前置處理器，來偵測中文手寫漢字的手寫角度，所以對於所消耗時間的量測與改善也是一個重要的因素，以下將要進行筆畫線段投票法與 K-mean clustering 處理速度的比較。我們用來測試的機器的處理器是 Intel Centrino 1.6GHz，配備 512 MB 隨機記憶體，運行 OS 是 Windows XP Home Edition，分別測試了沒有旋轉 (0)、旋轉 84.375°(30)、旋轉 168.75°(60)、旋轉 253.125°(90)，337.5°(120)等五組的樣本，每組各有樣本 77890 個。表 4-5-1 筆畫線段投票法與筆畫線段投票法+K mean clustering 處理速度比較表(單位:ms)顯示量測所得到的數據，中央的欄位所代表的意思，依照由左至右的順序非別是全部花費時間、前處理花費時間、前處理所花費時間占全部時間的百分比、以及平均辨識一個字所要花費的時間，時間的單位都是毫秒 (ms)。

數 角 度	方 法 據	筆畫線段投票法	筆畫線段投票法+K-mean Clustering
0(0°)		286994/ 5241/ 1.8%/ 3.68	3301774/ 3130591/ 94.8%/ 42.39
30(84.384°)		283649/ 3592/ 1.3%/ 3.64	3803625/ 3634297/ 95.5%/ 48.83
60(168.75°)		284950/ 3705/ 1.3%/ 3.66	3655139/ 3485436/ 95.4%/ 46.93
90(253.125°)		282063/ 3856/ 1.4%/ 3.62	3868845/ 3697551/ 95.6%/ 49.67
120(337.5°)		284597/ 3675/ 1.3%/ 3.65	3654567/ 3485976/ 95.4%/ 46.92

表 4-5-1. 筆畫線段投票法與筆畫線段投票法+K-mean clustering 處理速度的比較表(單位:ms)

從表 4-5-1 中我們得知，單獨使用筆畫線段投票法來偵測旋轉的角度，其速度符合我們原先的預期，占用整體的時間比率很低。而筆畫線段投票法加上 K-mean Clustering 所佔用的時間與前者則是完全相反，大部分的時間都是用來處理旋轉角度的偵測，不僅如此，整個的辨識時間還比筆畫線段投票法多出 11~12 倍。K-mean clustering 中使用了不少浮點數的運算以提高合併後的方向以及誤差評估函數的精確度，又需要多次反覆的重新計算，因此花費較多的時間並不令人意外，但是多出這麼多倍的時間則是始料未及的。



#### 4.6 實驗系統的成果展示

比較筆畫線段投票法與筆畫線段投票法+K-mean clustering 在旋轉角度偵測上的效果，我們發現後者的效果不僅比較好，而且穩定性也比較高，所以我們採用後者作為本實驗系統的主要偵測方法。圖 4-6-1. 實驗系統成果展示一號圖，與圖 4-6-2. 實驗系統成果展示二號圖記錄了筆畫比較複雜的字的偵測旋轉角度後校正的結果，虛線框住的黑色軌跡表示是使用者的輸入資料，虛線旁邊的紅色軌跡表示是系統校正之後的結果，軌跡第一畫起筆的地方是一個小圓點，在軌跡的重心的地方也有一個紅色的小圓點。

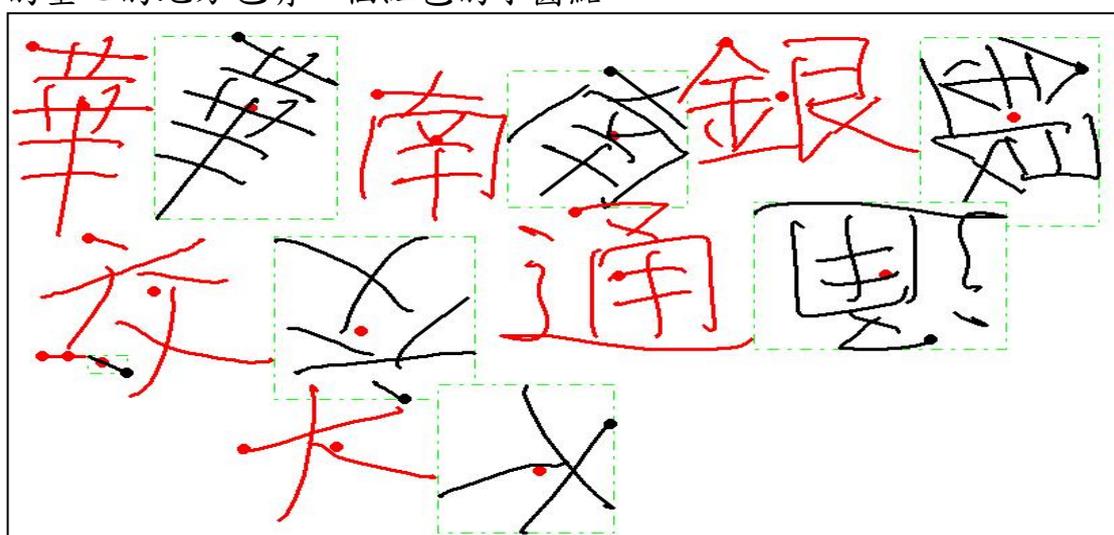


圖 4-6-1. 實驗系統成果展示一號圖

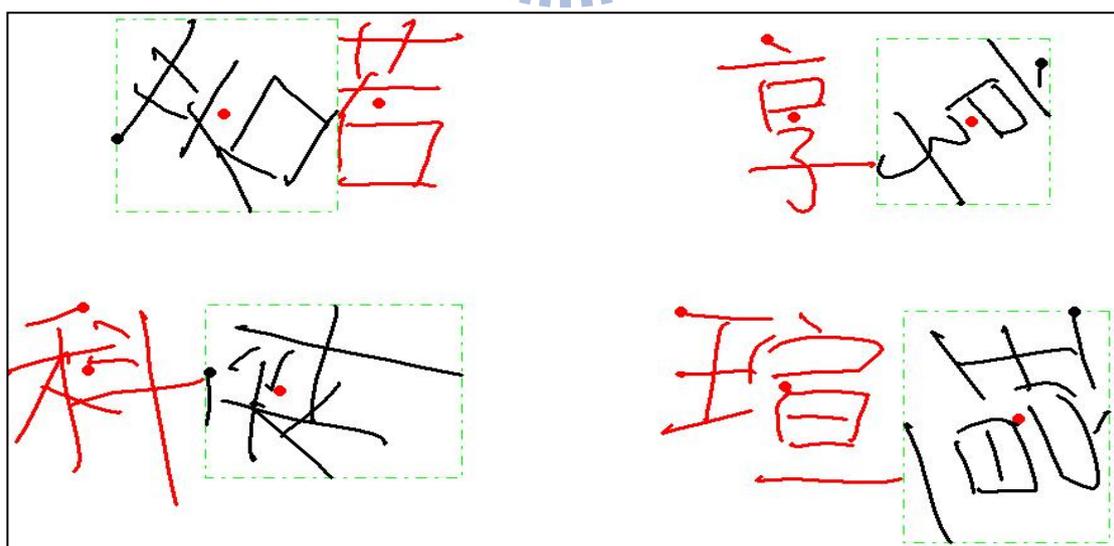


圖 4-6-2. 實驗系統成果展示二號圖

圖 4-6-3. 實驗系統展示三號圖顯示在沒有旋轉的時候，系統偵測旋轉角度的結果，當筆畫比較複雜的時候，不容易出現誤判斷的情況。但是圖 4-6-4. 實驗系統展示四號圖則可以看出筆畫比較簡單的字，在沒有使用經驗法則輔助的時候，很容易出現誤判，特別是不存在橫豎筆畫的情況下誤判率將近 100%。虛線圈住的黑色軌跡是使用者輸入的資料，虛線旁邊的紅色軌跡是校正後的結果。

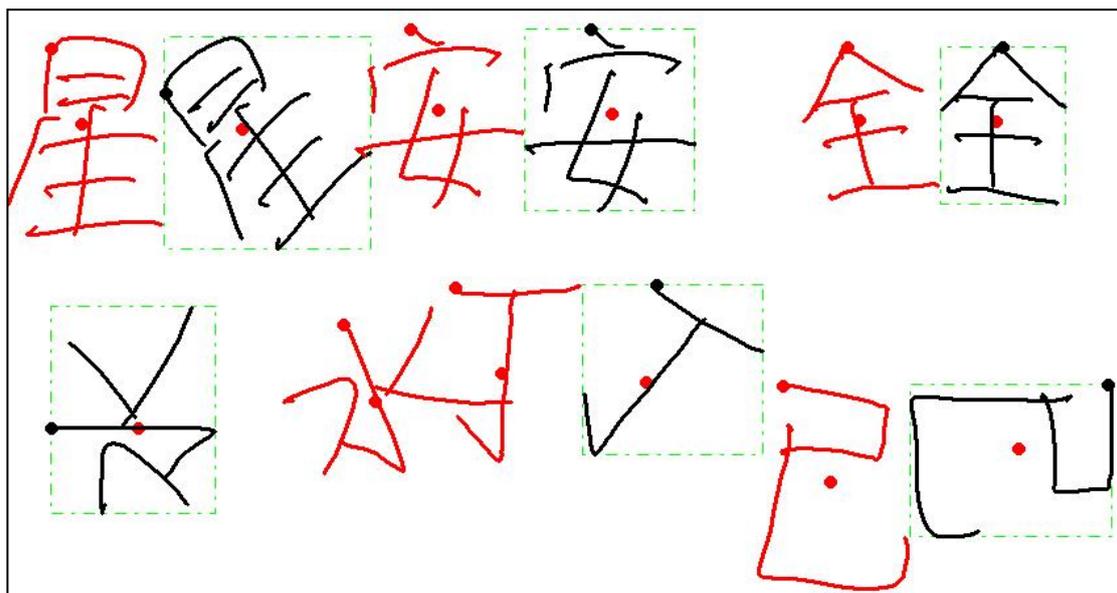


圖 4-6-3. 實驗系統成果展示三號圖

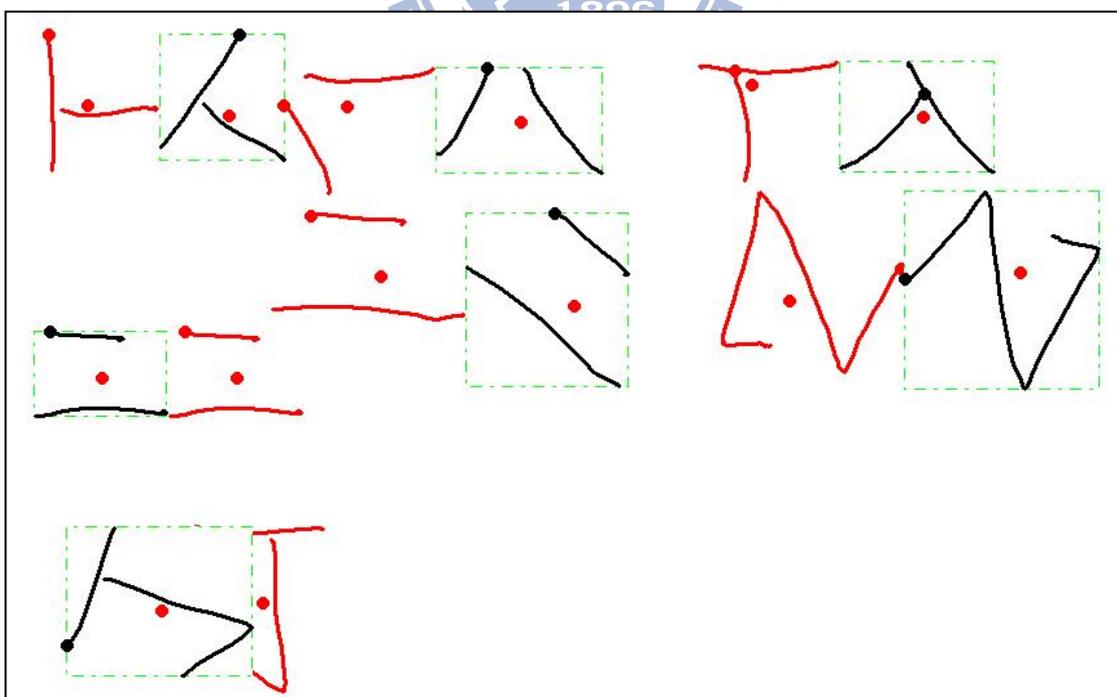


圖 4-6-4. 實驗系統成果展示四號圖

圖 4-6-5. 實驗系統成果展示五號圖顯示加入經驗法則輔助之後，對於筆畫簡單或者是缺乏特定橫筆或者豎筆的字，誤判的情形減少許多。虛線圈住的黑色軌跡是使用者輸入的資料，虛線旁邊的紅色軌跡是校正後的結果。

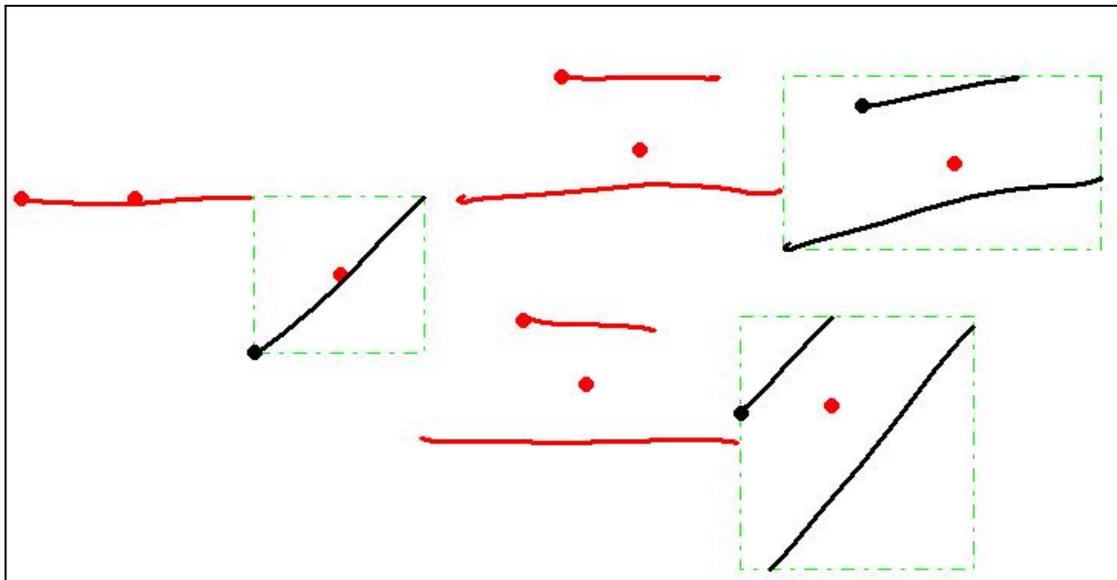


圖 4-6-5. 實驗系統成果展示五號圖



## 第五章 實驗結果討論及未來研究方向

### 5.1 實驗結果討論與分析

根據實驗結果顯示，我們的假設，橫筆線段與豎筆線段即使旋轉之後還是維持著相對位置的垂直關係，是正確的，使用筆畫線段投票法算法也有一定的成效，但是平均只能夠達到六成五左右的改善成果，特別是對於沒有旋轉的字進行測試的時候，還是有三成的誤判率。造成誤判的原因是：(1)樣本找到的線段其實是錯誤的，因為橫筆線段與豎筆線段的方向與長度變異較大，這些變異造成算法的誤判。(2)筆劃本身就違反了有垂直筆劃與水平筆劃的假設，雖然大部分的漢字都有者兩者，但是缺少其中一個方向，甚至兩個方向都沒有的字也是存在的，前者例如「一」、「三」、「川」等等，這些可透過經驗的法則來進行改善；後者例如「人」、「八」、「入」等等，這些即使應用經驗法則來加強，效果仍是相當有限。圖 5-1-1(a)(b)(c)(d)缺乏橫筆或者豎筆的漢字範例 就是這些字的代表，5-1-1(a) 只有豎筆，5-1-1(b) 沒有豎筆也沒有橫筆，5-1-1(c)只有一個豎筆，5-1-1(d)只有橫筆。

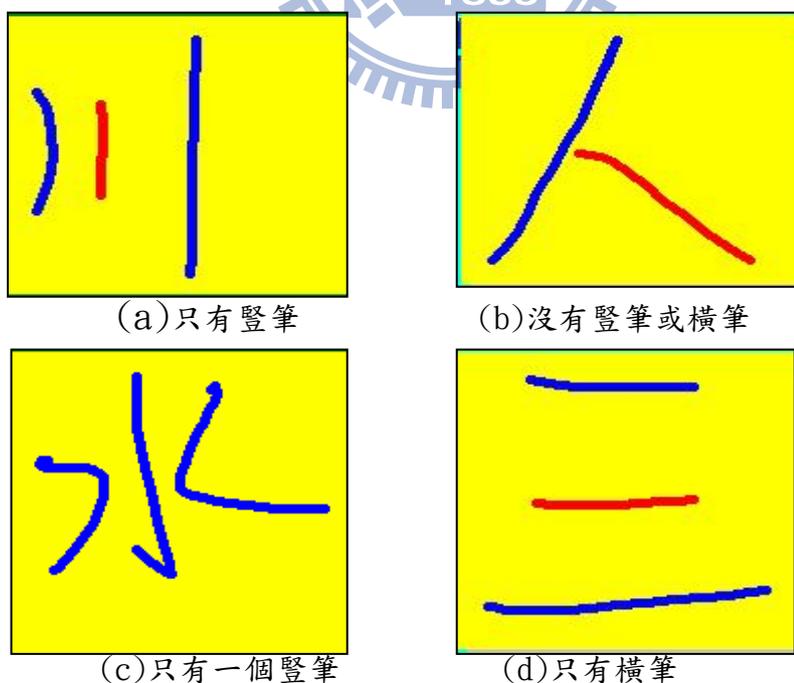


圖 5-1-1(a)(b)(c)(d). 缺乏橫筆或者豎筆的漢字範例

(3)對於方向與長度都相近的小線段無法合併來檢查，如果抖動的情況比較嚴重的話，則真正的方向可能反而會被其他數值比較大的線段隱藏起來，造成錯誤。

這種情況在引入 K-mean clustering 之後得到很大的改善，除了這個方法本身就可以利用群聚(cluster)的效應來合併周為相鄰的小線段之外，另外本論文所採用的新中心點計算方式卓有成效也是一個重要的原因。如果每一個線段方向所貢獻的權重一樣的話，反而會造成比單獨使用筆畫線段投票法差的效果，因為群聚之中可能會存在著方向近似但是長度很小的線段，這樣的線段對整體的貢獻很少，卻有機會被選為群聚的中心，無疑是一種錯誤的應用。所以本論文採計了線段的長度作為每個線段方向的權重，這樣長度比較大的方向就一定會被選為中心，或者中心會在這個線段方向的附近。

不管使用筆畫線段投票法或者是筆畫線段投票法加上 K-mean clustering 都會遭遇到一個問題， $359^\circ$ 與 $0^\circ$ 是相近的嗎？這個問題對於橫筆也就是水平向右的筆畫特別重要，因為使用者極有可能寫出在那附近擺盪的水平筆畫，必須適當處理邊界的情況才能夠正確的解決本論文的問題。我們的解法是：如果兩個線段方向的差距大於 $90^\circ$ ，就改取其補角就能夠解決這個問題。

## 5.2 未來研究方向

在本文提到的相關課題還是有許多方向可以嘗試，首先是關於筆畫線段投票法的部分，在原來的作法上可以再考慮對於合併與容錯規則的改善，因為這部分的分析都是需要人工介入的，不同時段不同人進行的分析可能得到不一樣的結果，其中或許有可以改善的部分。其次是針對 PCA 處理的部分，按照我們原先的設想，是嘗試以通過掃描輸入影像的方式，歸納出軌跡點分布的主軸，並且期待它就是所要找的校正方向前面的實驗證明這樣的作法是不穩定的，不過，還是可以

嘗試看看如果以筆畫的線段方向與長度作為統計的基礎，是否可以滿足所求。第三項是關於 K-mean clustering 的改進，可以針對初始中心的選取規則，相鄰角度的合併，新的中心點的計算方式以及迴圈次數的調整上進行實驗，找出真正的最佳解；更重要的是如何提升這個方法在旋轉偵測上的處理速度。第四項則是目前旋轉角度偵測的效果還是低於我們所預期的表現，而且對於沒有旋轉的字的誤判比率還是偏高了，所以提升偵測效果以及降低誤判比率是一個比較重要的方向，可以考慮整合參考文獻中的其他方法或者是另外找出新的應用規則。第四項是從實驗系統的結果來考慮，筆畫線段投票法用來輔助 K-mean clustering 解決無法偵測的角度效果不錯，對於「一」「二」的處理效果可以從圖上看出來，可以考慮增加對應的經驗法則來處理其他效果不佳的軌跡，並改進經驗法則的穩定性。



## 第六章 參考文獻

1. Tong-Hua Su, Tian-Wen Zhang, Hu-Jie Huang, Yu Zhou, Harbin Inst. of Technol., Harbin, "Skew Detection for Chinese Handwriting by Horizontal Stroke Histogram," 9th International Conference on Document Analysis and Recognition, Vol. 2, PP. 899-903, 23-26 Sept. 2007.
2. 魯湛與丁曉青, "基於筆段方向信息的聯機手寫漢字傾斜矯正算法," 模式識別與人工智能, 第13卷第4期, 2000年12月。
3. Stuart C. Hinds, James L. Fisher, and Donald P. D'Amato, Mitre Corp., McLean, VA, "A document skew detection method using run-length encoding and the Hough transform," 10th International Conference on Pattern Recognition, Vol. 1, PP. 464-468, 16-21 June 1990.
4. C. Verplaetse, "Inertial proprioceptive devices: Self-motion-sensing toys and tools," IBM Systems Journal, Vol 35, Issue 3-4, PP. 639-650, 1996.
5. Gatos, B., N. Papamarkos and C. Chamzas, "Skew detection and text line position determination in digitized documents," Pattern Recognition, Vol. 30, Issue. 9, PP. 1505-1519, 1997.
6. 曾逸鴻, 林裕淵, "中文文件影像中之特殊字體偵測," 科學與工程技術期刊, Vol. 3, No. 4, PP. 29-30, 2007.
7. 沈鴻光, "Character Recognition Using Nonlinear Normalization Algorithm and Fuzzy Nearest Prototype Classifier," M.S. thesis, 中原大學, 1994.
8. Rajiv Kapoor, Deepak Bagai, T. S. Kamal, "A new algorithm for skew detection and correction," Pattern Recognition Letters, Vol. 25, Issue 11, PP. 1215-1229, August 2004.
9. JONATHAN J. HULL, S.L. Taylor, Eds., "DOCUMENT IMAGE SKEW DETECTION: SURVEY AND ANNOTATED BIBLIOGRAPHY,"

10. Document Analysis Systems II, World Scientific, PP.40-64, 1998.
11. Masaki Nakagawa and Motoki Onuma, "On-line Handwritten Japanese Text Recognition free from Constrains on Line Direction and Character Orientation, "Proceedings of the Seventh International Conference on Document Analysis and Recognition, pp.519-523, 2003.
12. Cheng-Lin Liu, Stefan Jaeger, Masaki Nakagawa, "Online Recognition of Chinese Characters:The State-of-the-Art, " IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 26, NO. 2, February 2004.
13. Teng LONG and Lianwen JIN, "A Novel Orientation Free Method for Online Unconstrained Cursive Handwritten Chinese Word Recognition, "Proc. 19<sup>th</sup> International Conf. Pattern Recognition, 2008.
14. Shengming Huang, Lianwen Jin And Jin Lv, "A Novel Approach for Rotation Free Online Handwritten Chinese Character Recognition, "10<sup>th</sup> Convergence on Document Analysis and Recognition, pp.1136-1140, 2009.
15. Kai Ding, Lianwen Jin, Xue Gao, "A New Method for Rotation Free Online Unconstrained Handwritten Chinese Word Recognition: A Holistic Approach, "10<sup>th</sup> International Convergence on Document Analysis and Recognition, pp.1131-1135, 2009.
16. Lindsay I Smith, "A tutorial on Principal Components Analysis", 26 February, 2002.