以分群法為基礎的多環鍊碼應用於向量地圖壓縮之研究

林順喜
國立台灣師範大學資訊工程研究所
linss@csie.ntnu.edu.tw

陳克奇
國立台灣師範大學資訊工程研究所
hawk@csie.ntnu.edu.tw

摘要

近年來 PDA (Personal Digital Assistant)等行動裝置(mobile device)使用趨於普及，電子地圖因為可以結合旅遊與生活資訊而成為一種普遍的應用。FHM (Fibonacci, Huffman, and Markov) 壓縮法是一個字典基礎(dictionary-based) 的改良式多環鍊碼(modified multi-ring chain code)，但其為一個失真 (loss) 壓縮法，而其失真程度主要在字典(dictionary) 的設計，後來有研究者以 k-means 分群法改進其字典的設計方式，降低失真。本論文主要提出兩個方面的改良技巧。一、在將道路向量化後，把每一向量的起點置於二維座標平面的原點，可以得到一個分佈圖。運用部分逆序取向量法調整道路儲存的順序，可以使得此向量分佈更為緊密，增進分群後每一群內的相似程度，進而提高字典的品質。二、如果道路向量化的程序是採用每一點減去前一點座標的方式，則原有 k-means 分群法中 k-means 誤差的計算方式將無法代表整個地圖實際的失真值，透過調整 k-means 分群法的誤差計算公式，可以確保分群法最小化的是實際的地圖失真。此兩種方法都可在不增加地圖儲存空間大小的前提下，大幅改良道路圖的精確度。

關鍵詞：向量地圖壓縮、多環鍊碼、k-means 分群法

一、緒論

近年來因為 PDA (Personal Digital Assistant)等行動裝置(mobile device)在價格與應用軟體漸漸能夠符合一般消費者的需求而趨於普及，相關的應用隨之而起。電子地圖因為可以結合旅遊與生活資訊而成為一種普遍的應用，儲存在行動裝置上不僅攜帶方便，可以隨時查看，瀏覽時又可放大、縮小，還可以依各種需要製作包含不同資訊的地圖，應用非常廣泛。地圖上資訊的查詢與路徑的規劃都是更進一步的應用，如配合上全球定位系統(Global Position System)更可以作導航或軍事用途。目前有許多公司已做出實際產品，如英瑞德的 MobileMap 行動地圖[17]、應奇資訊的掌心地圖[19]、研勤科技的行動地圖系列產品[16]。因為硬體資源是行動裝置本身的限制之一[11]，如：記憶體較一般桌上型電腦小很多，所以，若要有效利用空間，勢必要壓縮地圖檔案。向量圖格式 (vector image) ，因為以點座標的方式儲存，所需空間比一般光柵圖 (raster image) 以儲存像素值方式還來得小。電子地圖若以線條為主的話，很適合使用向量格式記錄。但當地圖資料量大之時，還是需要對向量圖壓縮才能符合行動裝置的限制。

FHM (Fibonacci, Huffman, and Markov) [5][10] 壓縮法是一個字典基礎(dictionary-based) 的改良式多環鍊碼(modified multi-ring chain code)[3][4][8][14]，主要應用於線條壓縮，但其為一個失真 (lossy) 壓縮法，而其失真的程度完全在於字典(dictionary) 的設計優劣，先前已經有研究者以 k-means clustering[6][7][9]改進其字典的設計方式[12][13]，本研究主要目的在於提出一個改良的方法，可以在不增加儲存空間的前提下，進一步減少其失真。

本研究探討在既定的向量格式電子地圖下，如何減少壓縮過程的失真，至於從實體地圖數位化及向量化到向量地圖的過程，如：多少線條組成一段道路或哪些線條組成一段道路，則不是本論文討論的範圍，那些將作為本論文的原始輸入資料。向量格式電子地圖有的包含位相資料(topological data)，儲存的是地理空間的相鄰、交接關係，原本相鄰的區域不能在壓縮後變成不相鄰，這是屬於不能採用失真壓縮的部分，也不在本論文的範圍。

二、我們的改良方法

在前人的研究中[12][13]提出了兩種將道路轉成向量的方法，一種是 Delta(i,i-1)，如圖 2.1。另一種是在同一連續的道路上每一點都減去第一點所求出的向量，本篇中稱此種取向量法為 Delta(i,0)，如圖 2.2。在該研究中證明，在 Delta(i,0)下，給定適當的中心(center)初始值，經由 k-means clustering 造出來的字典必定都比 FHM 所造的失真還小。但同樣由

k-means clustering 造出來的字典，採用 Delta(i,i-1)所產生的失真必定比 Delta(i,0)還小。原因是 Delta(i,i-1)所產生的向量平均長度皆比 Delta(i,0)較短且集中，採用 Delta(i,0)所產生的向量則因為道路點越來越遠的關係，使得向量越來越長(見圖 3.2)，使得向量在二維平面上分佈得比較分散，所以分群後的 k-means error 較大，失真也就較大。本研究所採用的向量地圖壓縮法建造字典流程如圖 2.3 所示。

圖 2.1 取向量法 Delta(i,i-1)

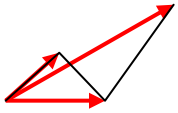

圖 2.2 取向量法 Delta(i,0)

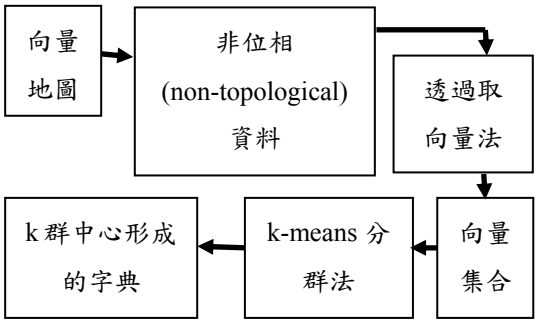

圖 2.3、改良式多環鍊碼壓縮法建造字典流程

第一節 部分逆序取向量法

由先前的敘述中可以看出壓縮失真完全取決於依分群結果所造出來的字典，而字典的品質除了改進分群演算法之外，還受到向量分佈的影響，一般的情形下，原始資料的向量分佈是無法改變的，但是在本研究中卻可以透過採用不同的取向量法來使得向量分佈更為集中，以增進字典的品質。因為前人的研究已經顯示出在同樣的條件下，採用 Delta(i,i-1)所造成的失真都會比 Delta(i,0)來得小，所以本研究主要針對 Delta(i,i-1)做改良。其關鍵方法就是改變道路點的記錄順序。假設一段道路由 (p_1, p_2, p_3) 三個點組成，採用 Delta(i,i-1)取得兩個向量為 $v_1 = p_2 - p_1$，$v_2 = p_3 - p_2$ 將其反序之後成為 (p_3, p_2, p_1) 仍然代表同一道路，但向量卻變為 $v_1' = -v_1$ $v_2' = -v_2$，如果 v_1, v_2 皆屬於第三象限，則 v_1'，v_2' 就都在第一象限。

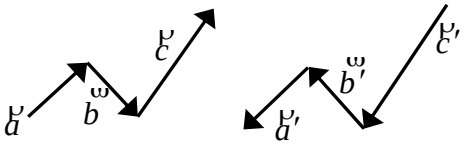

圖 2.4、逆序取向量

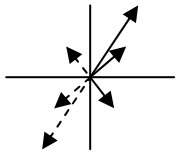

圖 2.5、逆序前後象限反轉

舉例來說。圖 2.4 中左右兩個道路是相同的一段道路，但右邊的路段經過反序，所以原本朝向第一象限 $\vec{a}$、$\vec{c}$ 的與朝向第四象限的 $\vec{b}$，分別變成朝向第三象限的 $\vec{a}'$、$\vec{c}'$ 與第二象限的 $\vec{b}'$ (如圖 2.5)。但這樣仍代表同一段道路。透過這樣一個過程，我們就可以把某兩個象限的點轉到另兩個象限，使得資料分佈更為集中。如圖 2.6, 2.7。橫軸為 X 軸，縱軸為 Y 軸。

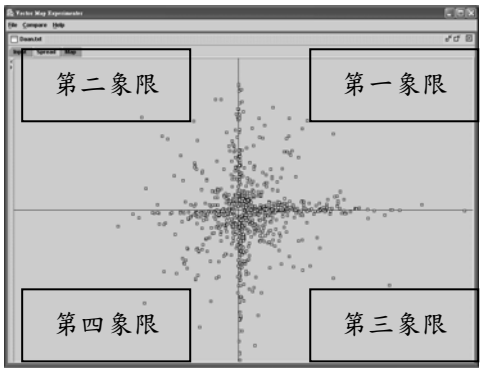

圖 2.6、逆序前台北市大安區道路向量分佈(採用 Delta(i,i-1))

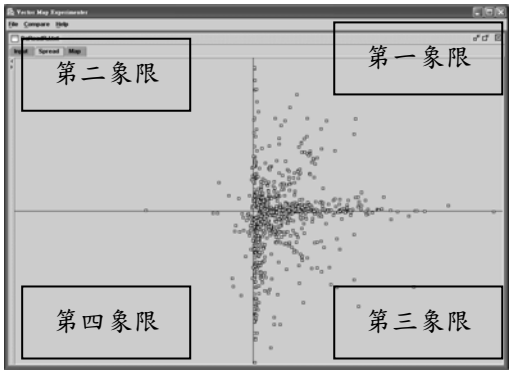

圖 2.7、逆序後台北市大安區道路向量分佈(採用 Delta(i,i-1))

這一個道路逆序的過程牽涉到兩個步驟，一、判斷要集中至哪兩個象限，在本論文中稱之「目的象限」，另兩個象限則稱「反序象限」，意指屬於其中的向量要被反序。二、判斷哪樣的道路要被逆序處理。本篇針對此兩

步驟都使用貪婪法(Greedy algorithm)來判斷。在第一個步驟中，為了效率考量，本研究比較地圖中相反象限的向量個數，也就是第一與第三象限比，第二與第四象限比，向量個數少的象限作為要被逆序的判定方向。在圖 3.6 中，一、四象限為目的象限，二、三象限為逆序象限。在第二步驟中，本研究比較該道路中，落在逆序象限與目的象限的向量個數，如果落在逆序象限的向量個數比落在目的象限的還多，那麼此段道路就有被逆序的價值，因此將其逆序。舉例來說，如果一段道路中，落在第三象限的向量有三個，落在第一象限的向量有一個，則此段道路就會被逆序。此方法如果要有集中向量分佈的效果需存在一個前提：大部分道路是循單一方向前進，即使有變化也不大。而這個前提多半是成立的，理由有二。一、人類社會的道路大多是朝固定方向進行。二、因為地圖向量化的演算法，如前所述，多半循線條追蹤，所以一條道路多半循既定方向記錄，所以方向不會差距過大。從圖 2.6 中也可以發現到並不是所有的向量都被轉到目的象限(第一及第四象限)。有一種道路會造成這樣的狀況：道路的向量同時分佈在反序以及目的象限。在此情形下，即使將整條道路逆序也無法將所有向量轉換至目的象限。因為逆序的過程會將屬於逆序象限與目的象限的向量互換，所以結果還是會存在有向量落在逆序象限。範例如圖 2.8。

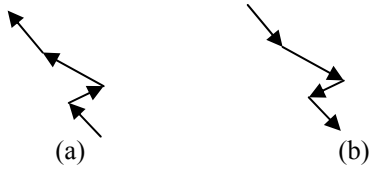

圖 2.8、例外道路逆序前後

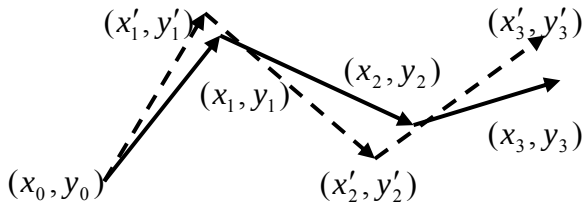

圖 2.9、實線為原道路，虛線為壓縮後道路

假設第一、四象限為目的象限，第二、三象限為逆序象限。圖 2.8(a)為逆序前的道路向量，有三段向量屬於第二象限(逆序象限)、有一段向量屬於第一象限(目的象限)，所以它會被逆序。而圖 2.8(b)為逆序後道路，有三段向量屬於第四象限(目的象限)，另一段向量屬於第三象限(逆序象限)。所以經過逆序後仍然無法將所有向量轉換至目的象限。

第二節 調整 k-means 分群法

本節將以誤差估計、向量的分配規則、空群處理與初始值給定四方面來說明本研究所提出的 k-means 分群法。

誤差估計。在先前研究中[13]已經證明在 Delta(i,0)下，k-means error 就代表了壓縮後地圖與實際地圖的誤差(real error)，所以最小化 k-means error 就等於最小化實際地圖的誤差，一般 k-means 分群法以移動至最近的群作為降低 k-means error 的手段。但在 Delta(i,i-1)下，k-means error 並不能代表實際誤差，所以單純以降低 k-means error 為目標，並不能有效的降低地圖的實際誤差。其原因如下：在圖 2.9 中，實線代表原有道路，其座標為 (x_i, y_i)；虛線代表壓縮後道路，其座標為 (x_i', y_i')，i=0~3。令 $\Delta x_i=x_i-x_{i-1}$, $\Delta y_i=y_i-y_{i-1}$，假設$(\Delta x_i,\Delta y_i)$屬於第 i 群，其中心為(Cx_i,Cy_i)，也就是$(\Delta x_i,\Delta y_i)$以(Cx_i,Cy_i)編碼。在 Delta(i,i-1)下，這整條道路所產生的 k-means error 為：

$$[(\Delta x_1-Cx_1)^2+(\Delta y_1-Cy_1)^2]+[(\Delta x_2-Cx_2)^2+(\Delta y_2-Cy_2)^2]+[(\Delta x_3-Cx_3)^2+(\Delta y_3-Cy_3)^2] \quad \text{--- (1)}$$

但實際地圖上壓縮前後兩點座標的距離平方和是

$$[(x_1-x_1')^2+(y_1-y_1')^2]+[(x_2-x_2')^2+(y_2-y_2')^2]+[(x_3-x_3')^2+(y_3-y_3')^2] \quad \text{--- (2)}$$

因為$(\Delta x_1,\Delta y_1)$屬於(Cx_1,Cy_1)，所以壓縮後的點 $x_1' = x_0 + Cx_1$ (y_1' 同理可得)代入方程式 (2) 的第一項，原式變成 $(x_1 - x_0 + Cx_1)^2 = (\Delta x_1 + Cx_1)^2$ (y 的部分同理可得)，這讓方程式(2)的第一項等於方程式 (1) 的第一項，但從第二個向量之後 $x_2' = x_1' + Cx_2$ 兩式就不再相等了。其主因就是沒有考慮前一段向量造成的誤差對往後向量的影響，以致這情況下 k-means error 並無法代表字典對實際地圖造成的誤差，所以最小化 k-means error 並不等於最小化地圖的實際誤差。因為壓縮後的座標可以由群的中心計算出來，如：$x_2' = x_0 + Cx_1 + Cx_2$ (y_2' 同理可得)，所以方程式(2)可以在演算法執行的過程中計算出來，為了使誤差更進一步能有幾何意義，所以本研究將(2)每一項都開根號，使之成為歐幾里德距離，如下式：

$$\sqrt{[(x_1-x_1')^2+(y_1-y_1')^2]}+\sqrt{[(x_2-x_2')^2+(y_2-y_2')^2]}+\sqrt{[(x_3-x_3')^2+(y_3-y_3')^2]} \quad \text{--- (3)}$$

如此即可代表字典實際造成的誤差，其幾何意義就是「地圖中所有點壓縮前後的幾何距離總和」。本論文將這種計算實際誤差的分群法稱為「累積誤差分群法(prefix k-means clustering)」。每一條道路內的向量要從頭至尾依序分群，累積誤差的估計才會正確。因為要計算前面向量所累積的誤差，如果前面的向量還沒有決定分配到哪一群，也就是還沒有決定壓縮後的值，後面的道路計算累積誤差時就無法參考前面向量的壓縮值，如果採用了，稍後前面的向量若被分配至新的一個群中，先前計算的累積誤差就不正確了。

向量的分配規則。既然 k-means error 不能代表實際誤差值，而將向量分配給最近的群只能確保減少 k-means error，但並不一定會減少實際誤差，所以必須比較該向量在新的群及舊的群所造成的實際誤差大小，以此決定是否分配至新的群。此時若再運用一個技巧，也就是同時尋找第二近、第三近的群(或更多)，看分配到哪一群可以減少最多的實際誤差，如果都不能降低，就不動。這樣必定能降低實際誤差，這可以在後面的實驗結果中看到。在本篇中稱此方法為「多鄰近群找尋」。

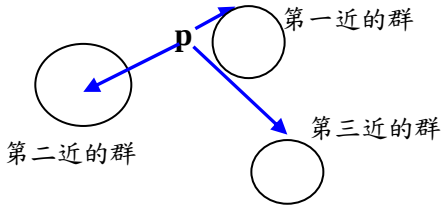

圖 2.10、多鄰近群找尋

上圖中，菱形點 p 為某一道路向量(在二維平面以點表示)，旁邊三個圓代表距離它最近的三個群，圓點為各群的中心。箭頭長度代表該點到各群中心的距離。如果鄰近群找尋的個數為 3，則 p 會先分配到第一近的群，也就是以第一近的群中心編碼，並計算分配到此群所產生的實際誤差。接著換第二、第三近的群，並進行同樣的計算，最後比較三者所產生的實際誤差，將 p 分配給產生最小實際誤差的群。不過這個方法如果鄰近群數量越來越多時，有時會讓 k-means 分群法不容易收斂到一個固定的分群狀態。因為如果只能移到最近的群，而最近的群不能降低誤差，就不會移動，這比較會讓分群的狀態固定下來。越多群可以移動，可能會讓點在幾個群之間跳動而無法固定下來。但在還有空群時，則直接將向量分配到最近的一群即可。採用此法有兩個原因：一、因為本法乃由空群去分攤產生最大誤差之群的點，一群點本來屬於一群，被分成兩群後誤差一定會下降，所以將向量直接分配到最近的一群可以省去許多計算累積誤差的時間。二、作者透過實驗發現，在處理空群時，使用實際誤差來作為是否重分配向量的依據並不會產生較好的結果。

空群處理。這裡的空群是指沒有被分配到任何一個點的群。但這在 k-means 分群過程中是常常發生的，不過通常發生在初期，因為群的中心初始值用一些經驗法則來給定，通常不一定能符合資料的分佈，這時就會有一些群的周圍沒有較近的點。所以在內迴圈中，我們將空群的中心強制設為目前最大實際誤差的群中的某一點，然後進行分配，這樣會使得該空群與最大誤差的群共同分攤那一群點，總誤差便會降低。假設圖 2.11 中橢圓形代表在所有群中，產生最大實際誤差的群，圓點代表該群的中心。此處所指之實際誤差，就是前面誤差計算部分所提到的，考慮每個向量之前的向量所造成的累計誤差。左方一個單獨的圓點代表沒有任何一點屬於該群，也就是空群。

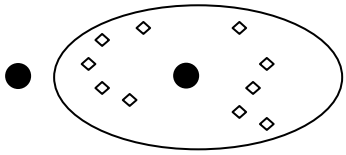

圖 2.11、產生最大實際誤差的群

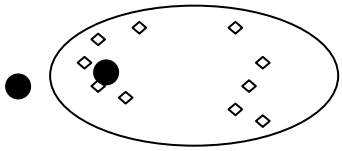

圖 2.12、設定空群中心

依前面所述要將空群的中心強制設為目前最大實際誤差的群中的某一點，結果會如圖 2.12。接著重分配向量至最近的群，分配結果會如圖 2.13。點與中心有線相連代表屬於該群。新加入的中心會吸引靠它較近的一些點，其他較遠的點就仍然會被分配到原本的中心。接下來每一群會根據所屬的點求出新的中心，如圖 2.13。結果這一群點就被拆成兩個更集中的群，誤差自然就降低。

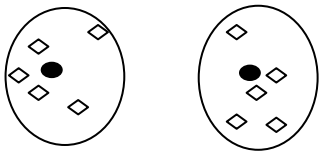

圖 2.13、各群新的中心

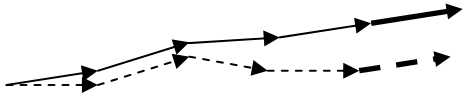

圖 2.14、後面的向量受前面向量影響

處理空群的其中一個步驟是要找到產生實際誤差最大的一個群，並以空群來拆解它，

但此時因為採用實際誤差，可能產生一個群只有一個成員但卻造成最大的實際誤差的狀況，此原因乃一條道路中向量較多時，較後面的向量因為前面向量累積的誤差太大造成。因為該群只有一個成員，這樣一來無法被另一空群拆解，二來搬到哪一群都會造成最大的實際誤差。舉例來說，如圖 2.14，上方實線為原始道路向量，下方虛線為壓縮後失真道路向量。假設最後一段向量壓縮得很精準、沒有誤差，但是因為前四段向量都有誤差，造成此段道路最後一段向量所造成的實際誤差(實際道路與壓縮後道路的最後一點距離)反而是所有五段向量中最大的，如果恰巧最後一段向量是某一群的唯一成員，而這一群的誤差又恰好是所有的群中最大的，則前述所提到利用空群去拆解誤差最大的群的方法在此會行不通。解決方式為取實際誤差值次之或更後但其成員卻不只一個的群。

初值選定。本論文以 FHM 字典的分佈方式作為各群的中心初始值，以輸入資料中最短的向量長度作為字典的單位距離。

完整演算法描述：下面將前述提到的分群演算法以假碼(pseudo code)作一完整描述：

```
1)給定 k 個群的中心一個初始值(此時所有的
      群都是空的)
2)當新的實際誤差總和≠舊的實際誤差總和{
3)      舊的實際誤差總和←新的實際誤差總
              和(←代表賦值)
4)      當 k 群中仍有空群時{
5)          將空群的中心設為目前實際誤差
              最大的群中的某一點
6)          將向量分配至最近的群
7)          計算各群中心
8)          加總各群內向量產生之實際誤差，
              並找出實際誤差最大的一群
9)          }
10)     依能否降低實際誤差來分配向量至各
            群中
11)     計算各群中心
12)     加總各群內向量產生之實際誤差，並
            找出實際誤差最大的一群
13) }
```

假設有 n 個向量，分成 k 群，多重鄰近群找尋的個數為 c。以下分析各步驟所需的時間複雜度：第七步：O(nk)。第三、四、八、九步：O(n)。(此部分與一般 k-means 分群法複雜度相同)。第二步：除了與每一群計算距離的時間外，還要加上與多個鄰近群計算實際誤差的時間，而推算實際誤差的時間長度與道路向量個數成正比，假設計算一個向量所造成的實際誤差所需時間為 O(t)，本步驟需 O(n(k+ct))。為了與先前的研究者比較，作者也在此描述對照用的一般 k-means 分群法：

```
1) 新的實際誤差總和≠舊的實際誤差總和{
2)      舊的實際誤差總和←新的實際誤差總
            和(←代表賦值)
3)      分配向量至最近的群中
4)      計算各群中心
5)      計算各群內之誤差總合，並找出誤差最
            大的一群
6)      如果還有空群時 {
7)          將空群的中心設為目前誤差最大
              的群中的某一點 }
8)  }
```

實作時作者還採用了一個技巧來加速上述演算法的速度，這個方法可以用在各式 k-means 分群法，就是在計算各群中心與誤差值時，只計算那些成員有增減的群即可，不必重新計算所有的群。就一般的分群問題來說，k-means 分群法令人詬病之處就是要人為給定 k 值，因為不知道要給定怎樣的值才能有比較好的分群效果，所以有些分群法會在分群的過程中調整群的數目。不過此項缺點在本研究中卻是一項優點，因為如果要針對不同需求產生出精準度不同的地圖時，可以用調整 k 值來做到，一般來說，k 值越大，失真會越小。

三、實驗與討論

本節將以實驗數據說明本研究中所提出兩項演算法的改進策略，有助於實際誤差的降低。為了不失一般性以及親切感，本研究使用台北縣市幾個區的地圖作為實驗輸入，使用 shapefile 格式[2][15]，座標系統為台灣地區二度分帶[1]。本實驗所需程式以 Java 語言撰寫，在 Intel Pentium III（851MHz）處理器、256MB 主記憶體、WindowsXP 下執行。

甲、部分逆序取向量法

接下來的圖表中，正序的數值代表前人的方法[12][13](使用一般 k-means 分群法)所建造的字典所產生的誤差，逆序的數值(本研究所提出之方法)代表輸入向量經過部分逆序取向量法後，再以一般的 k-means 分群法建造字典所產生的誤差。以上兩種方法只差在有無使用部分逆序取向量法，作者在不同的分群數下比較兩者誤差值。由以下圖表可看出部分逆序取向量法能夠有效的改進字典品質、降低誤差，尤其在分群數較少時。而分群數越多，兩個方法的誤差都會降低且越來越接近。表 3.1 列出執行所需時間，以秒為單位。逆序最多只需耗費 O(n)的時間，所以實際上不會增加很多

時間，而 k 值越大，時間就越久。

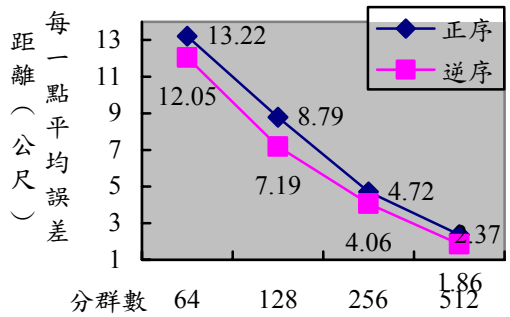

圖 3.1、台北市中山區逆序正序誤差比較

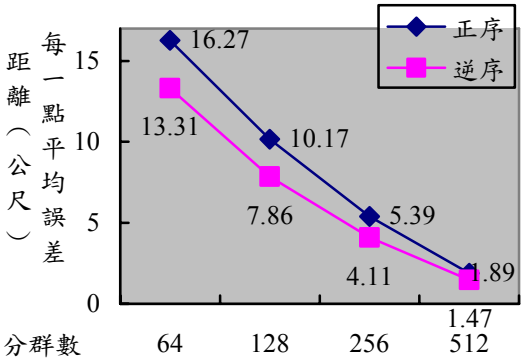

圖 3.2、台北市大安區逆序正序誤差比較

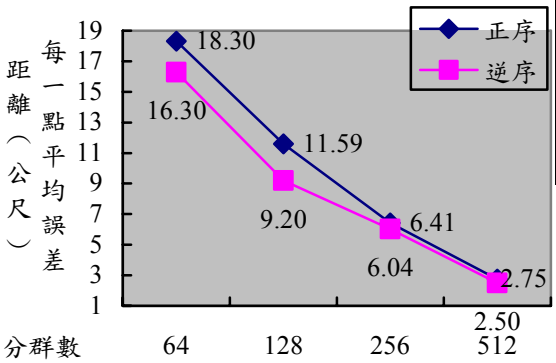

圖 3.3、台北市信義區逆序正序誤差比較

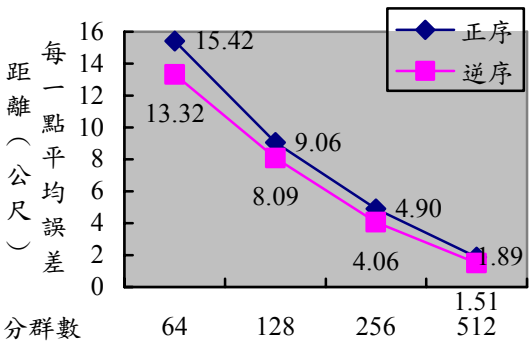

圖 3.4、台北市松山區逆序正序誤差比較

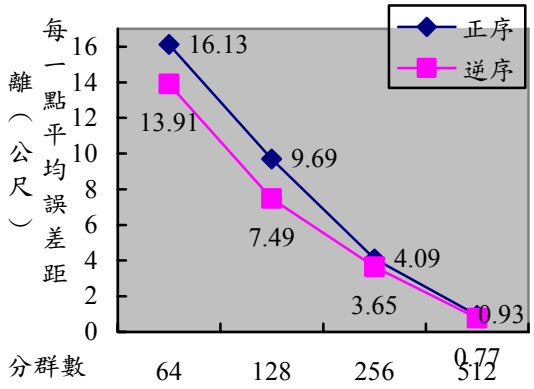

圖 3.5、台北縣永和市逆序正序誤差比較

表 3.1、正序逆序執行時間比較

分群數	64	128	256	512
台北市中山區(向量個數：1422)				
一般	2	12	54	174
逆序	3	10	48	314
台北市大安區(向量個數：1000)				
一般	1	8	31	106
逆序	1	8	34	264
台北市信義區(向量個數：1347)				
一般	2	9	43	145
逆序	2	8	44	332
台北市松山區(向量個數：966)				
一般	3	7	34	163
逆序	1	7	32	278
台北縣永和市(向量個數：707)				
一般	1	4	23	135
逆序	1	4	24	96

乙、累積誤差分群法 (prefix k-means clustering)

此處本研究以前人研究成果[11][12](使用一般 k-means 分群法)與累積誤差分群法做比較，但不使用部分逆序取向量法。在圖表中分別以 k-means 和 Prefix(**i**)代表，**i** 代表採用「多鄰近群找尋」技巧時，所找的鄰近群個數。由以下圖表可以得知，如果單只運用累積誤差分群法的話，並不能保證得到較小的誤差，多數得到比一般 k-means 分群法稍高一點的值，偶爾較低。但如果配合「多鄰近群找尋」就能確保誤差更小了。這原因是累積誤差分群法在分配向量至各群時，是以"能否降低實際誤差"作為移動至新一群的基準，但最近的群往往不能降低實際誤差，反而使得向量無法移動到別的群，使得結果反而比一般 k-means 差，所以如果多幾個鄰近的群可以移動，將有助於實際誤差的降低。表 3.2 列出執行時間，以秒為單位。

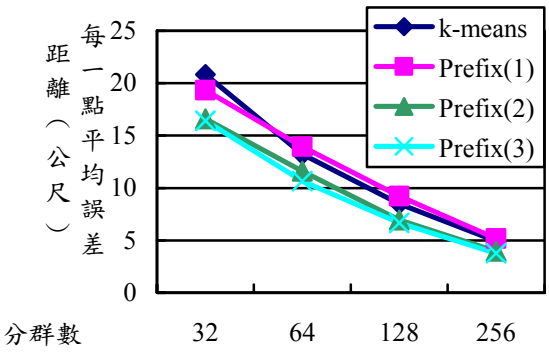

圖 3.6、台北市中山區一般 k-means 分群法與累積誤差分群法誤差比較

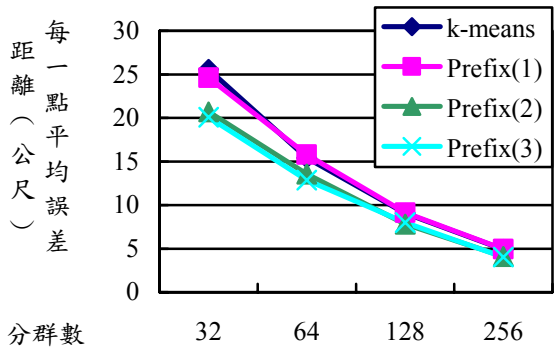

圖 3.9、台北市松山區一般 k-means 分群法與累積誤差分群法誤差比較

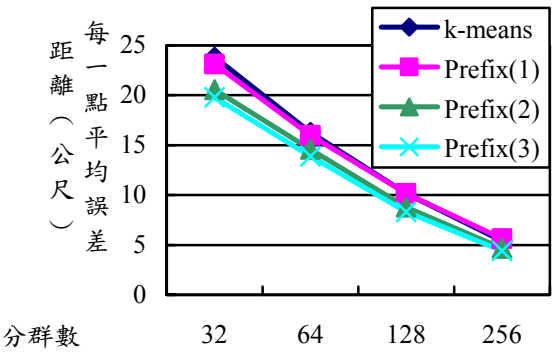

圖 3.7、台北市大安區一般 k-means 分群法與累積誤差分群法誤差比較

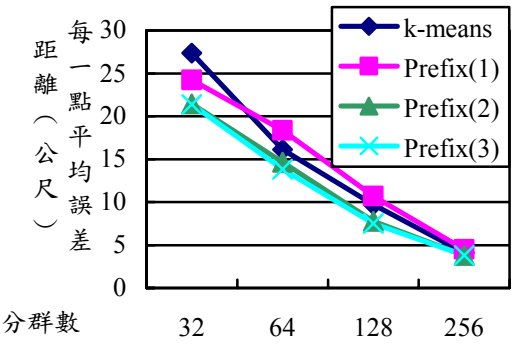

圖 3.10、台北縣永和市一般 k-means 分群法與累積誤差分群法誤差比較

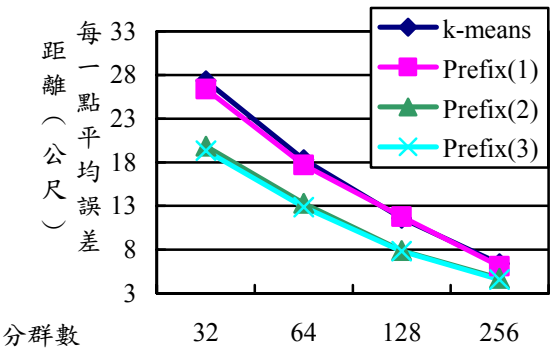

圖 3.8、台北市信義區一般 k-means 分群法與累積誤差分群法誤差比較

因為 k-means 分群法很容易受輸入資料分佈的影響，即使資料量相同，所需執行時間仍可能有很大的差異，所以此處所列之執行時間，不能很精準的代表所有同樣大小的地圖，而「多重鄰近群找尋」也會讓執行時間增加。

表 3.2、累積誤差分群法在找尋鄰近群數目不同下的執行時間比較

分群數	32	64	128	256
台北市中山區(向量個數：1422)				
Prefix(1)	3	3	11	35
Prefix(2)	14	9	23	60
Prefix(3)	138	58	176	116
台北市大安區(向量個數：1000)				
Prefix(1)	1	5	14	114
Prefix(2)	3	37	9	38
Prefix(3)	16	20	22	46
台北市信義區(向量個數：1347)				
Prefix(1)	3	6	25	55
Prefix(2)	67	131	431	264
Prefix(3)	182	281	238	194
台北市松山區(向量個數：966)				
Prefix(1)	1	2	6	29
Prefix(2)	5	4	10	195
Prefix(3)	14	35	36	261
台北縣永和市(向量個數：707)				
Prefix(1)	1	1	3	29
Prefix(2)	3	8	9	114
Prefix(3)	15	10	150	72

因為先前的學者[12][13]使用美國(共 52 州)地圖[18]作為實驗輸入，其檔案格式同為 shapefile，但座標為經緯度，共有 99333 個向量。本研究也採用同一網站之道路圖，以一般 k-means 分群法及本研究提出的演算法做比較，群數為 256 群，而誤差的計算方式也沿用前人的計算方式，乃是將每一點原始座標值與壓縮後的座標值相減的平方之總和。原始的經位度座標 (x_i, y_i)，$i = 1 \sim 99333$，(x_i', y_i') 為壓縮後的座標，則誤差總和計算式如方程式(4)。由表 3.3 其誤差總和可以看出本研究所提出之改良演算法可以得到較低的失真。

$$\sum_{i=1}^{99333} (x_i - x_i')^2 + (y_i - y_i')^2 \qquad \text{---(4)}$$

表 3.3、美國道路圖一般 k-means 分群法與累積誤差分群法誤差比較

	誤差總和	執行時間(秒)
一般 k-means 分群法	11.04	1831
本研究提出之分群法(多重鄰近群找尋個數為 3)	4.14	102255

四、結論與未來發展

在本研究所實驗的地圖中，在取向量法上，運用部分逆序取向量法都得到比較好的結果。未來也可以嘗試將較長的向量用較短的兩個(或多個)向量組合而成，這可以讓在二維座標上較外圍的一點變成較靠中心的兩點(或多點)，好處是可以集中向量分佈，壞處是會增加壓縮後儲存所需的空間，因為原本一個碼可以代表的向量變成需要兩個碼。而累積誤差分群法則要配合「多鄰近群找尋」才會得到比一般 k-means 分群法還小的誤差。但 k-means 分群法的缺點是會受到各群中心初始值的影響，不同的初始值下可能產生不同的分群結果，而這會影響字典的品質，所以未來可以嘗試利用其他分群法來解決此一問題。

五、參考文獻

[1] 周天穎,地理資訊系統理論與實務,台北：儒林圖書,民92年.

[2] ESRI Shapefile Technical Description, July 1998.

[3] H. Freeman, “On the encoding of arbitrary geometric configurations”, Institute of Radio Engineers, Transactions on Electronic Computers, 1961.

[4] H. Freeman, A. Saaghri, “Generalized Chain Codes for Planar Curves”, International Conference on Pattern Recognition, 1978.

[5] J. D. Gibson, T. Berger, T. Lookabaugh, D. LindBergh, R. L. Baker, Digital Compression for Multimedia Principles & Standards, Morgan Kaufmann; 1st edition, 1998.

[6] J. A. Hartigan, M. A. Wong, "A k-means clustering algorithm", Applied Statistics, No.28, pp.100-108, 1979.

[7] A.K. Jain, Dubes R.C., Algorithms for clustering data, Prentice Hall, 1988.

[8] B. Johannessen, J. H. Bons, N. B. J. Weyland, R. Prasad. “Multiring Differential Chain Codes for Line Drawings.” Electronics Letters, Vol. 26, No.10, 1990.

[9] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, A. Y. Wu. “An Efficient k-Means Clustering Algorithm: Analysis and Implementation.” IEEE Transactions on Patterns Analysis and Machine Intelligence, Vol. 24, No. 7, 2002.

[10] D. Salomon, Data Compression: the Complete Reference, Springer-Verlag, 2nd ed., 2000.

[11] M. Satyanarayanan, “Fundamental Challenges in Mobile Computing ”, Fifteenth ACM Symposium on Principles of Distributed Computing, 1996.

[12] S. Shekhar, Y. Huang, and J. Djugash, “Dictionary Design Algorithms for Vector Map Compression(Abstract).” In Proceedings of Data Compression Conference, 2002.

[13] S. Shekhar, Y. Huang, J. Djugash, C. Zhou, “Vector Map Compression: A Clustering Approach”, ACMGIS 2002, 2002.

[14] Annotated Computer Vision Bibliography, http://iris.usc.edu/Vision-Notes/bibliography/contents.html.

[15]Arc/Info. http://www.esri.com/software/index.html.

[16] Maction Mobile Technologies 研勤科技, http://www.mactiontech.com/index.htm.

[17] MobileMap, http://www.gismosoft.com.

[18] National Atlas of the United State of America - Map Layers Warehouse http://nationalatlas.gov/atlasftp.html.

[19] TeleMap, http://www.telemap.com.