國立交通大學

多媒體工程研究所

碩 士 論 文

結合模糊與可能性叢集法之基於樣板的叢集演算法之分析

Analysis of Shell Clustering Algorithms for Template-Based Shapes that Combine Fuzzy and Possibilistic Clustering Approaches

研 究 生：劉強

指導教授：王才沛 教授

中華民國九十八年八月

結合模糊與可能性叢集法之基於樣板的叢集演算法之分析

Analysis of Shell Clustering Algorithms for Template-Based Shapes that Combine Fuzzy and Possibilistic Clustering Approaches

研 究 生：劉強　　Student：chiang liu

指導教授：王才沛　　Advisor：Tsaipei Wang

國 立 交 通 大 學

多 媒 體 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

August 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年八月

結合模糊與可能性叢集法之基於樣板的叢集演算法之分析

學生：劉強　　指導教授：王才沛

國立交通大學多媒體工程研究所 碩士班

摘要

本篇的論文目的在於探討資料分群的結果，特別地，我們想要研究fuzzy c-means (FCM)和possibilistic c-means(PCM)的影響，並且組合他們，在基於樣板的shell clustering上。基於樣板的shell clustering是執行特殊幾何形狀偵測的叢集演算法。使用在shell clustering上的FCM和PCM曾發表過許多研究。然而，FCM和PCM有他們的缺點。例如，FCM的結果容易被雜訊影響，而PCM易於產生重疊的群。

我們特別感興趣的，即是在探討將 FCM 和 PCM 演算法組合過後，是否能對 shell clustering 的分群達到更好的成果。在此我們引用了兩個文獻上的組合演算法，possibilistic fuzzy c-means (PFCM) 和 improved possibilistic c-means (IPCM)。在實驗結果中，發現到混合性的叢集演算法 PFCM 和 IPCM 套用在樣板理念上後，在偵測複雜圖形或是雜訊資料時，較 FCM 和 PCM 來的有益，我們也發現到不同的混合性叢集演算法含有不同的特性，在做叢集分群時能夠更有幫助。

Analysis of Shell Clustering Algorithms for Template-Based Shapes that Combine Fuzzy and Possibilistic Clustering Approaches

Student：chiang liu　　Advisor：Tsaipei Wang

Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

Abstract

This goal of this thesis is to investigate the results of data clustering. Specifically, we want to study the effect of fuzzy c-means (FCM) and possibilistic c-means (PCM), as well as their combinations, in template-based shell clustering. Template-based shell clustering is the process of detecting clusters of particular geometrical shapes through clustering algorithms. The use of FCM and PCM in shell clustering has appeared in many research. However, both FCM and PCM have their shortcomings. For example, the results of FCM are highly affected by noise, and PCM tends to produce overlapping clusters.

We are particularly interested in whether the combination of FCM and PCM algorithms can improve the results of shell clustering. Here we use two combinational algorithms in the literature, possibilistic fuzzy c-means (PFCM) and improved possibilistic c-means (IPCM). Our results indicate that IPCM and PFCM have better shape detection results than FCM and PCM when used with template-based shell clustering of complex or noisy data. We also discover that different combination methods have different properties that are helpful in clustering.

誌謝

這篇論文能夠順利完成，首先要感謝的，是我的指導教授王才沛老師，對於不是資工本科系升上來的我，在我遇到研究上的問題或者是困難時，老師都會耐心的指導以及教誨，並且給予我適當的建議或是正確的解決方向，讓我對於論文的研究更加的順利，非常感謝老師的栽培。感謝陳玲慧老師以及楊敏生老師擔任我的口試委員，提供許多寶貴的建議，使得本論文更加完備，特此致謝。接下來要感謝的是實驗室一起研究與努力的夥伴們。和我一起口試的昇毅和耿維，有 了你們的陪伴，讓我寫論文的過程不會感到孤獨，在心情低落或是緊張時，也能互相打氣。也要感謝俞邦、偉誌和 崇桂，有 了你們，讓我的碩士生活更加充實和精彩，最後我要感謝我的家人，謝謝你們能夠支持我完成碩士學業，沒有你們的支持與鼓勵，就不會有今日的我。

目錄：

圖例：

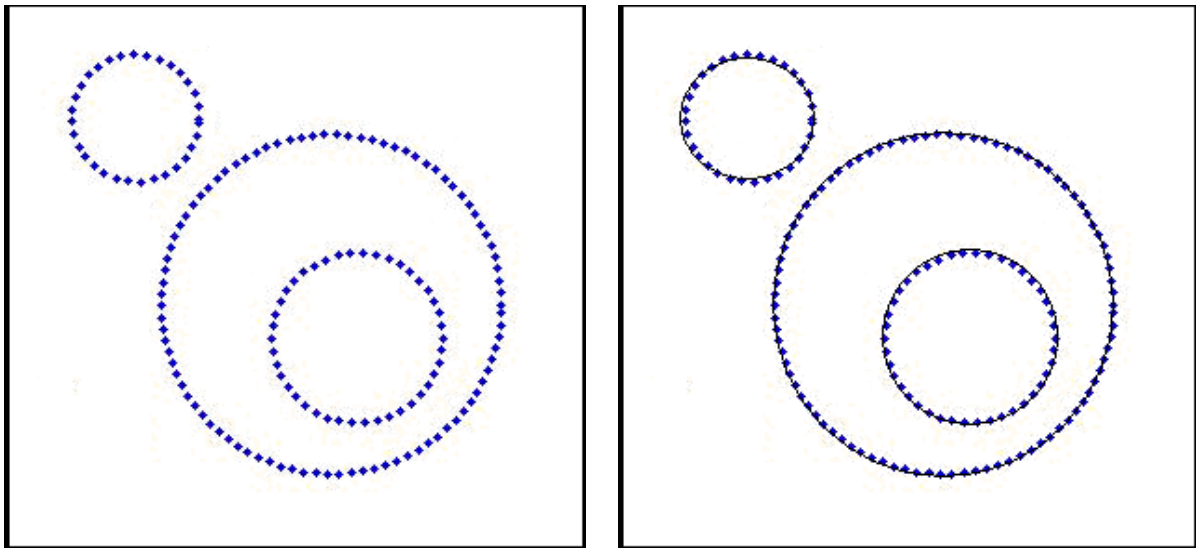
表格：

第一章：簡介

1.1 研究動機

叢集分析(cluster analysis)，主要用意爲探討資料與資料之間是否擁有關聯性，並且藉由這些關聯性將他們分群[1]、[2]。分群過後，同一群的資料彼此之間會較爲相似，反之不同群的資料差異性就會很大。這樣的應用範圍很廣，包括醫學、圖學和生物學等。

由於資料愈來愈複雜，在偵測辨識上不能只單靠距離相近而判斷爲同一群，可能在資料的形態上，有著幾何形狀的分佈情形，他們可能會以圓形或是方形等的分佈來表達彼此的關連性，因此針對幾何形狀的叢集分析也相當重要。此類分群，我們稱之爲shell cluster，分群不再只是搜尋叢集的中心點位置，而是一個形狀，如圖1。圖1(a)爲資料的分佈，此處爲三個圓形。在圖1(b)中，即爲叢集偵測完成的結果，各自找到了三個不同的圓圈。從圖中我們可以發現，分群不再只是搜尋叢集的中心點位置，而是一個形狀。



(a) (b)

圖1：shell-shaped cluster的分群概念 (a)資料分群前 (b)分群後的資料

在1989年Dave[3]首先提出fuzzy c-shell(FCS)的叢集演算法，使我們可以針對圓形或橢圓形的資料叢集類型做偵測，之後，又可利用FCS演算法來偵測影像上的曲線[4][5]，而adaptive fuzzy c-shells cluster algorithms[6]又可以改善FCS演算法的效能。但這些演算法的充其量也只是針對圓形或弧形的叢集做偵測，對於其他的形狀卻沒有解決的辦法。雖然也有科學家設計出針對矩形的偵測方法[7]，但是仍然無法套用至所有的形狀。爲了一組新的資料集而設計一個新的解決方法實在太過於麻煩，因此樣板(Template)的理論[8][9]便開始受到重視，在進入叢集演算法時，先設計出希望偵測的目標圖形，將圖形放入叢集演算法，由此便可以找出跟所給圖形相似的資料集合，大大的解決了千變萬化的資料集合分群問題。但是對於shell cluster的偵測，與point prototypes相比，錯誤的機率都會大幅上升，可能會偵測到錯誤的資料，或者是形狀很相似但是不正確的資料，因此在演算法的搭配上，就變成了一項重要的議題。

在演算法的選擇上，起初是採用fuzzy c-means(FCM)[10]和possibilistic c-means(PCM)[11]兩種”prototype-based clustering”的演算法做分析。FCM不會使得叢集造成重複，但是FCM的問題在於，很容易受到雜訊資料點影響，使得叢集的中心點位置不精準，這樣的問題，在shell cluster中，更是容易受到影響。而PCM的目的，在於改善分群上容易受到雜訊影響所產生的問題，但也容易受到起始位置影響，偵測到重複的叢集。Possibilistic fuzzy c-means (PFCM)[12]和improved possibilistic c-means (IPCM)[13]兩者爲混合FCM和PCM的叢集演算法，基本上將兩者混合的用意在於，FCM和PCM的能力和缺失剛好相反，希望藉由混合達到互補的結果。在PFCM用一個加權值來區分FCM和PCM所含的比率，然後再將FCM的membership和PCM的typicality相加混合，藉由給予的加權值不同，能套用的叢集分群也會更廣，使得偵測上更富有彈性。IPCM最初的目的在於改PCM容易產生叢集重複的問題，希望加入fuzzy的因子，將重複的可能

性降低，方法基本上在於將計算好的PCM typicality，乘上額外計算的FCM membership，達成混合的效果。

在本篇論文中，即是在研發將FCM和PCM混合後的叢集演算法PFCM和IPCM，套用在樣板基礎上，產生兩組新的shell cluster分群演算法，並且分析這樣嶄新的架構與以往以FCM和PCM爲基礎的shell cluster演算法的不同。

1.2 論文結構

本篇論文共分爲五章。在第一章中，將簡單說明何謂叢集分析及其目前所遭遇的問題。第二章爲文獻探討，先簡述FCM和PCM兩種叢集演算法的特性，並針對shell cluster的叢集演算法和樣板做說明，章節最後探討混何性叢集演算法的目的還有未來的發展。第三章則是將混合性叢集演算法套用在樣板理論的步驟。第四章則對於混合性叢集演算法做分析。最後第五章是本論文的研究心得以及未來之研究方向。

第二章：文獻縱覽

由於資料特性的不同，也代表著分群的複雜性，要如何找出一種叢集演算法可以應用於各種資料集上，仍是目前科學家所在研究的。而目前使用較廣泛且較有概觀性的分群法則中，比較有名也較常被使用的就是c-means法則[1]。c- means詳細的演算法如下：首先起始條件是輸入欲分類的資料群N個，以及需求的叢集數C。接著將N分配給距離最近的叢集，當全部的N都分完後，再重新計算各個叢集，也就是取該樣本群中的平均值，即爲新的叢集中心點，如此反覆做下去，直到滿足收斂條件爲止。此外，資料集與叢集的距離有相當多種的算法，最基本的作法就是計算歐幾里得距離(Euclidean distance)。

雖然 c-means 有著簡單的計算與快速收斂的優點，但缺點是分群的中心相當易受起始位置的影響，也就是通常只能尋找到區域的最佳解。Fuzzy c- means(FCM)[10]的法則在於資料點 x 不會絕對地屬於任何叢集，而是以一個介於 0-1 之間的數字來表示 x 隸屬於某個叢集的程度，而且將每個資料點 x 對於不同叢集的 membership 值加起來會得到 1。但如果資料的雜訊過多，FCM 會使得某資料點雖然實際上爲雜訊，但對於不同叢集的 membership 相加仍會得到 1，故對於叢集中心點的移動產生影響力，而造成叢集中心的錯誤。Possibilistic c- means(PCM)[11]便可以解決這類雜訊問題，它與 FCM 不同的是，資料點不是相對的去跟每個叢集做比較，資料點 x 離某叢集愈近，則它的距離就會愈短隸屬於此群的程度就會愈高。PCM 採用一個叫做 η 值的參數，它的用途在於掌控每個叢集的偵測範圍，避免不正確的情形發生。舉例來說：資料群 A 和資料群 B，叢集在偵測的時候，因爲雛型離 A 比較近，因此會忽略了 B 的存在，但是可能 B 群才是叢集所要搜尋的正確資料，爲了避免這樣的情形發生，加入 η 值能讓較遠的資料點不容

易被忽略，然而 PCM 所存在的問題，在於分群結果容易受初始位置所影響，因爲資料點 x 對於不同叢集的 membership 加起來不一定是 1，因此叢集與叢集間不會知道對方的偵測情況，容易造成叢集的偵測重複。

對於 shell cluster 的研究，Dave 所提出 fuzzy c-shall(FCS)[3][4] 叢集演算法，藉由半徑扣除叢集與資料點的距離，使我們可以針對圓形的資料分佈做偵測。隨後又提出了針對橢圓形的叢集類型做偵測的[5][6][14][17]，也有針對矩形的資料形態作偵測[7]，此時不但要計算圖形的位移量，也要注意圖形的旋轉。[7]中後半段所提到的 2-Rectangular，是藉由將兩個矩形重疊，所產生的橢圓形偵測法。對於降低分群的計算量，也有學者著手研究，論文[16]即爲改善 FCS 的一個例子。諸如此類解決 shell cluster 的方法很多，但對於過於複雜的形狀，即便是上面的演算法，也沒有辦法偵測完全。

爲了解決更複雜的shell cluster問題，此時樣板[8][9]就相當的重要，在初始便輸入一種特定的形狀，之後演算法便按照此種形狀去做偵測，因此藉由輸入形狀的多樣化，所能偵測的形狀也更多，在[9]中還可以將此形狀放大縮小和旋轉更可以將之變形扭曲以達到更高的辨識效率，但是由於無法利用單一中心點來詮釋整個圖形，所以樣板的計算量會較其他演算法來的龐大。

由於 FCM 和 PCM 演算法的能力以及缺失剛好相反，因此有許多的學者便開始著手於 FCM 和 PCM 的混合理論，希望能夠使得優缺點互補，對於偵測分群達到更好的效益。Fuzzy-possibilistic c-means(FPCM)[15]即爲一種混合性叢集演算法，它的混合理念在於將 FCM 和 PCM 所求得的 membership 和 typicality 值做相加，所得的值視爲 FPCM 的 membership 值。然而這樣的做法有一個需要探討的問題，相加過後的值如果超過 1 會不會對分群有所影響，因此 possibilistic fuzzy c-means(PFCM)[12]便因此產生。PFCM 是由 Pal 等人所研發出來的一種新的叢集演算法，主要宗旨在於改善 FPCM 的問題，在

混合 FCM 和 PCM 的 membership 和 typicality 值時，同時利用 a 和 b 兩值來區別不同的 membership 所佔的比例，此一改善不但解決了 FPCM 本身的問題，還可以依據更改 a 和 b 值讓演算法更富有彈性，能夠運用在不同的需求上。Improved possibilistic c-means(IPCM)[13]也爲一種混合性叢集分群演算法，它本身的目的在於改善 PCM 演算法所帶來的問題，主要是分群容易受初始位置影響和叢集容易重疊，所以希望在求得 PCM 的 typicality 值後，乘上一個額外計算的 FCM membership 值做爲調和，改善 PCM 的問題。然而 PFCM 和 IPCM 在往後也常常被其他學者做引用，例如將 PFCM 套用在 kernel method[18]，方便解決高維度的分群，或是把 IPCM 和 PFCM 做比較分析的對象[19][20]，藉此得知自身演算法的偵測效率。

第三章：將各類演算法套用至樣板理念

以下我們所討論的樣版架構，以及叢集的轉換，皆是採用2008年的“Possibilistic Shell Clustering of Template-Based Shapes”[9]爲理念基礎。

3.1 點爲雛形的C-means叢集演算法

3.1.1 FCM和PCM

下面爲常見的FCM目的函式[10]：

$$J = \sum_{j=1}^{C}\sum_{i=1}^{N} u_{ij}^{m} d_{ij}^{2} \tag{1}$$

N爲資料點個數，C爲叢集個數，m是模糊化因子，u_{ij}爲資料點 x_i 在叢集 θ_j 上的membership值，這裡 x_i 爲第i個資料點座標(1≦i≦N)， θ_j 爲第j個叢集(1≦j≦C)的參數，最後 d_{ij} 爲 x_i 和 θ_j 的距離。這裡FCM的membership有一限制

$$\forall i, \quad \sum_{j=1}^{C} u_{ij} = 1 \tag{2}$$

爲了求解 u_{ij} 和 θ_j 的方程式，必須要尋找(1)的區域最小值，因此對(1)做偏微分，求解 $\partial J / \partial u_{ij} = 0$ 和 $\partial J / \partial \theta_j = 0$ 便可以獲得兩者的方程式，下面爲 u_{ij} 和 θ_j 的方程式

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{d_{ij}}{d_{ik}}\right)^{\frac{2}{m-1}}} \tag{3}$$

$$\theta_j = \frac{\sum_{i=1}^{N} u_{ij}^m x_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{4}$$

下面爲即將要探討的PCM目的函式[11]：

$$J = \sum_{j=1}^{C}\sum_{i=1}^{N} h_{ij}^n d_{ij}^2 + \sum_{j=1}^{C} \eta_j \sum_{i=1}^{N} (1-h_{ij})^n \tag{5}$$

h爲PCM的typicality值。PCM的目的函式跟FCM的很相似，但是PCM不採用FCM的限制(2)，因此在解區域最小值時，不能利用到(2)，要用到新的參數η_j，此時的η_j可視爲 叢集 對每個資料點的偵查範圍，因此typicality的方程式更改如下

$$h_{ij} = \frac{1}{1+\left(\frac{d_{ij}^2}{\eta_j}\right)^{\frac{1}{n-1}}} \tag{6}$$

由方程式可知，當資料點與叢集距離愈近時，d 值會愈小，h 的值就會愈大，屬於此叢集的可能性就會愈高。

FCM 和 PCM 的演算法如下：

步驟一：選擇$\theta_j(0)$爲初始的 叢集 中心θ_j的位置， j ＝ 1 …C

固定 m 值 (或 n 值)

步驟二：初始 T＝０　(T 代表迴圈計數，計算迴圈的執行次數)

步驟三：更新 membership 值(3) (或 typicality 值(6))

步驟四：更新叢集中心位置(4)

步驟五：T=T+1

步驟六：如果滿足$\| \theta(T) - \theta(T-1) \| < \varepsilon$則結束，否則將回到步驟三，

其中 ε 爲一極小數

$\| \theta(T) - \theta(T-1) \| < \varepsilon$ 的涵義在於，計算第 T 次和第 T-1 次的叢集中心位置兩者的距離，是否小於一個極小值，如果是則表示叢集中心收斂到固定位置，如果不是則需要繼續執行。

3.1.2 混合性叢集演算法

混合性叢集演算法這裡採用的是 PFCM 和 IPCM，兩種的目的函式都以(1)和(5)做修改。下面爲 PFCM 的目的函式[12]：

$$J = \sum_{j=1}^{C}\sum_{i=1}^{N}(au_{ij}^{m} + bh_{ij}^{n})d_{ij}^{2} + \sum_{j=1}^{C}\eta_j \sum_{i=1}^{N}(1-h_{ij})^{n} \tag{7}$$

此時的 u_{ij} 和 h_{ij} 分別代表的涵義爲FCM和PCM的membership和typicality值，a和b爲兩個大於0的值，用於控制FCM和PCM的混合比率，n跟m一樣，是模糊化因子。之後求解(7)的區域最小值，即可獲得PFCM的 u_{ij} 和 h_{ij} 還有 θ_j 的方程式

$$u_{ij} = \frac{1}{\sum_{k=1}^{C}(\frac{d_{ij}}{d_{ik}})^{\frac{2}{m-1}}} \tag{8}$$

$$h_{ij} = \frac{1}{1+(\frac{bd_{ij}^{2}}{\eta_j})^{\frac{1}{n-1}}} \tag{9}$$

$$\theta_j = \frac{\sum_{i=1}^{N}(au_{ij}^{m} + bh_{ij}^{n})x_i}{\sum_{i=1}^{N}(au_{ij}^{m} + bh_{ij}^{n})} \tag{10}$$

而IPCM的目的函式如下[13]：

$$J = \sum_{j=1}^{C}\sum_{i=1}^{N}(u_{ij}^{m})[h_{ij}^{n}d_{ij}^{2} + \eta_j(1-h_{ij})^{n}] \tag{11}$$

u_{ij} 和 h_{ij} 分別代表的涵義爲 FCM 和 PCM 的 membership 和 typicality 值，在最外圍乘上 FCM 的 membership 使其混合。對(11)做偏微分所得的 u_{ij} 和 h_{ij} 還有 θ_j 方程式爲

$$h_{ij} = \frac{1}{1+(\frac{d_{ij}^{2}}{\eta_j})^{\frac{1}{n-1}}} \tag{12}$$

$$u_{ij} = \frac{1}{\sum_{k=1}^{C}[\frac{(h_{ij})^{(n-1)/2}d_{ij}}{(h_{ik})^{(n-1)/2}d_{ik}}]^{\frac{2}{m-1}}} \tag{13}$$

$$\theta_j = \frac{\sum_{i=1}^{N}(u_{ij}^{m}h_{ij}^{n})x_i}{\sum_{i=1}^{N}(u_{ij}^{m}h_{ij}^{n})} \tag{14}$$

混合性叢集演算法也跟FCM和PCM的做法相似，藉由一再更新membership的值來計算叢集新的位置，直到做到最佳解，不同處在於每次迴圈中皆要先算出 u_{ij} 和 h_{ij} 的值 ，才可計算 θ_j 。

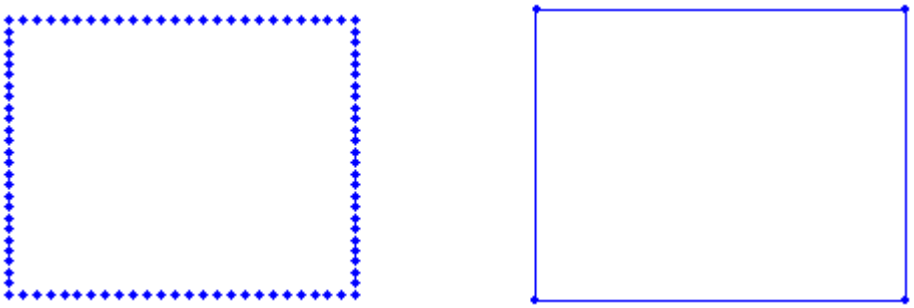
3.2 基於樣板的shell clustering

在shell cluster下，每個叢集不再只是一個點，因此最先要探討的，是如何計算資料點和叢集的距離 d_{ij} ($=d(x_i,\theta_j)$)。以下，我們引用[9]來做演算法架構。首先，我們要先來分析樣版的型態，主要可以區分爲兩種，分別是以點爲基準和以邊爲基準。以點爲基準是的用點來描繪出所要偵測的形狀，而以邊爲基準則是利用頂點跟邊來描繪出圖形的形狀，如圖3-1。本篇論文中，都是採取以邊爲基準來做分析，因此叢集演算法所求

的點到群的距離 $d(x_i,\theta_j)$，這裡即是計算點到 θ_j 中最近的邊的最短距離，如下

$$d_{ij} = d(x_i,\theta_j) = \| x_i - p_{ij} \| \tag{15}$$

p_{ij} 爲資料點 x_i 到叢集 θ_j 的最短距離座標，如果超過了邊的範圍而求不出最短距離，就改爲計算點到 θ_j 中最近的頂點的距離，如圖3-2



(a) (b)

圖3-1：樣板的型態 (a)以點爲基準 (b)以邊爲基準

(a) (b)

圖3-2：資料與叢集的距離 (a)點到邊的最短距離 (b)點到頂點的距離

再來要作定義的，是樣板的轉換

$$P_j = H(T;\theta_j) \tag{16}$$

這裡H代表一個轉換函式，P_j 爲第j個轉換過後的叢集，而T爲一個集合，所代表的涵義，即爲這個樣板所涵蓋的點和邊，我們亦可將轉換方乘式寫成

$$p = H(p^*;\theta_j) \tag{17}$$

此時 p 和 p^* 皆爲 P_j 上的同一個點，只是相對應轉換後和轉換前的位置

之後要探討的，便是如何做轉換，這裡採用[9]的typeI方法，保留shell cluster的原型，不做圖形的扭曲或是伸縮，只針對叢集的位移量、大小收縮和旋轉。其轉換方式如下

$$p = H(p^*;\theta_j) = R_j S_j p^* + t_j \tag{18}$$

其中 R_j、S_j 和 t_j 分別爲 P_j 的旋轉、大小收縮和位移量的數值，其中 R_j 爲一個 2×2 的旋轉矩陣由旋轉角度 φ_j 所組成

$$R_j = \begin{pmatrix} \cos\varphi_j & -\sin\varphi_j \\ \sin\varphi_j & \cos\varphi_j \end{pmatrix} \tag{19}$$

瞭解了樣板的定義方式後，便可逐步套入不同的叢集演算法，開始做分析。

3.3 將樣板套用至FCM、PCM、PFCM和IPCM上

起初的目標，我們必須先得到 S_j、t_j 和 φ_j 的方程式[9]，我們先以FCM爲主要架構，再往下延伸。我們知道 p_{ij} 爲 θ_j 上面的其中一個點，藉由typeI的轉換可以由(18)式將(15)改爲

$$d_{ij} = d(x_i,\theta_j) = \| x_i - p_{ij} \| = \| x_i - (R_j S_j p_{ij}^* + t_j) \| \tag{20}$$

之後將(20)代入(1)中，即可獲得樣板的目的函式 J ，對 J 做 S_j、t_j 和 φ_j 偏微分等於零做求解，我們即可獲得下列式子

$$\frac{\partial J}{\partial t_j} = \sum_{i=1}^{N} (2u_{ij}^m)(R_j S_j p_{ij}^* + t_j - x_i) = 0 \tag{21}$$

$$\frac{\partial J}{\partial S_j} = \sum_{i=1}^{N} (2u_{ij}^m)(R_j S_j p_{ij}^* + t_j - x_i)^T (R_j p_{ij}^*) = 0 \tag{22}$$

$$\frac{\partial J}{\partial \varphi_j} = \sum_{i=1}^{N} (2u_{ij}^m) S_j (R_j S_j p_{ij}^* + t_j - x_i)^T (\frac{dR_j}{d\varphi_j} p_{ij}^*) = 0 \tag{23}$$

之後再求解(21)-(23)的等式，可以得到

$$t_j = \frac{\sum_{i=1}^{N} u_{ij}^m (x_i - R_j S_j p_{ij}^*)}{\sum_{i=1}^{N} u_{ij}^m} \tag{24}$$

$$S_j = \frac{\sum_{i=1}^{N} u_{ij}^m (x_i - t_j)^T (R_j p_{ij}^*)}{\sum_{i=1}^{N} u_{ij}^m \| p_{ij}^* \|^2} \tag{25}$$

$$\varphi_j = \tan^{-1}\left[\frac{\sum_{i=1}^{N} u_{ij}^m (x_i - t_j)^T \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} p_{ij}^*}{\sum_{i=1}^{N} u_{ij}^m (x_i - t_j)^T p_{ij}^*}\right] \tag{26}$$

m是模糊化因子，u_{ij} 爲資料點 x_i 在叢集 θ_j 上的membership值，此時是取FCM的membership值計算方式(3)代入

Fuzzy Shell Clustering of Template Based shapes演算法如下

步驟一：選擇 $\theta_j(0)$ 爲初始的 叢集 中心 θ_j 的位置， j ＝ 1 …C

固定 m 值

步驟二：初始 T＝ 0

步驟三：利用(20)計算距離 d_{ij} 再利用 d_{ij} 和(3)計算 membership 值

步驟四：利用(24)(25)(26)分別計算出 t_j 、 S_j 和 φ_j

由 φ_j 可藉(19)計算出 R_j

步驟五：T=T+1

步驟六：更新每一個 叢集 型態

$$\theta_j(T) = R_j S_j \theta_j(T-1) + t_j$$

步驟七：如果滿足 $\| \theta(T) - \theta(T-1) \| < \varepsilon$ 則結束，否則將回到步驟三，

其中 ε 爲一極小數

PCM、PFCM和IPCM與FCM不同處在於membership值計算的不同，以及要引用的η值該如何做處理。因爲η值的涵義在於定義一個理想的偵測範圍，如果搜尋的範圍太小可能會忽略較遠的資料，範圍太大可能會將所有的資料分爲同一群，因此理想的搜索範圍應該由大而小慢慢收斂，逐步將叢集移到正確的位置。因此本篇論文的 η_j 調節方法爲[9]

$$\eta_{j(T)} = \max[\eta_{\min}, r_\eta \eta_{j(T-1)}] \tag{27}$$

在這裡 $\eta_{\min}$ 爲一最小值，而 r_η 爲一倍數乘積，用以降低 η_j 值($0 < r_\eta < 1$)， $\eta_{(0)}$ 視爲一個初始且最大的值($\eta_{(0)} \approx$ [資料的總範圍/10]^2)。而PCM的membership計算方式採用(6)。

Possibilistic Shell Clustering of Template Based shapes 演算法和Fuzzy Shell Clustering of Template Based shapes 演算法的差異在於，步驟二時要初始 $\eta_{(0)}$ 值，步驟三要用(6)計算 typicality 值並且要更新 η_j 值(27)，步驟四計算 t_j 、 S_j 和 φ_j 時 u_{ij} 值由 h_{ij} 值代入。

對於PFCM來說，membership的混合採用FCM membership和PCM typicality分別乘上一倍數做相加，代入(8)和(9)，整體的membership = $au_{ij}^m + bh_{ij}^n$ ，a和b爲介於0到1的參數。

因此計算叢集的型態轉換 R_j 、 S_j 和 φ_j 時，只要將 u_{ij} 改爲 $au_{ij}^m + bh_{ij}^n$ 即可。IPCM同理，只要將membership和typicality相乘，以混合過後的membership $= u_{ij}^m h_{ij}^n$ 代入。

Possibilistic Fuzzy Shell Clustering of Template Based shapes和Improved Possibilistic Shell Clustering of Template Based shapes的演算法流程跟Possibilistic Shell Clustering of Template Based shapes的一樣，只是在步驟三要同時計算 u_{ij} 和 h_{ij} 的值，並且計算出整體的membership，之後代入的 u_{ij} 便是整體的membership值。設計好了各類叢集演算法，我們便可以開始討論這幾種演算法對於shell cluster的偵測有甚麼不同效果。

第四章：混合性叢集演算法的分析

在這裡我們將針對雜訊還有容易重複辨識的影像做實驗，探討混合性叢集演算法可否順利解決FCM和PCM的問題，再來便針對初始叢集的個數來討論混合性叢集演算法的效率，最後則是針對η值做分析。

對於shell cluster來說，主要用於偵測資料的形狀分佈，因此首先要探討的，是何謂正確的偵測。首先，我們先由範例圖4-1來定義何爲正確的分群，不但要偵測到正確的形狀(如圖4-1(a)資料爲方形分佈)，而且雛型的邊緣不能離資料太遠，如圖4-1(b)，在偵測範圍內的資料視爲同一叢集，範圍外的則不是。這裡給予的偵測範圍爲資料與資料間的距離大約0.4至0.6單位長度。因爲演算法最後會依據叢集邊緣一定的範圍將資料點取下，因此如果離邊緣太遠或者是給予的範圍太廣，可能會造成資料點擷取不正確或是擷取到多餘的資料點。圖4-1(c)爲正確的偵測，資料皆在偵測範圍內，圖4-1(d)爲錯誤的偵測，資料超出了給予的偵測範圍。

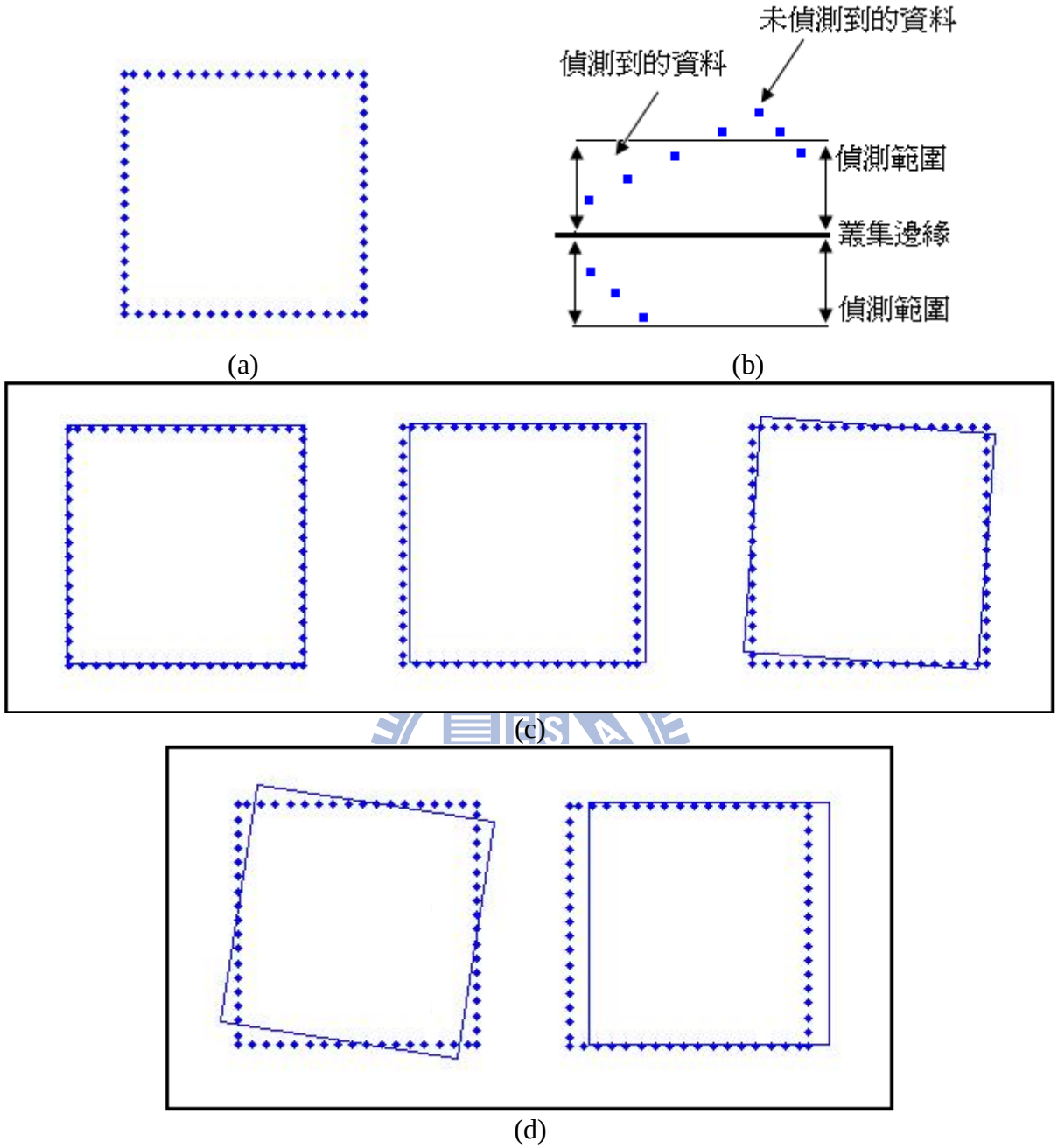


圖4-1：叢集偵測示意圖 (a)尚未分群的資料 (b)叢集偵測範圍 (c)正確偵測的範例 (d)偵測不正確的範例

4.1 雜訊的影響(noise)

我們知道混合性叢集演算法的目的在於解決改善FCM和PCM的問題，因此本章節的研究目的在於藉由雜訊資料的偵測來分析混合性演算法所的辨識率，是否比FCM來的精確。我們利用下面資料來做分析(圖4-2)，圖4-2(a)初始圖中爲三個正方形，正方形彼此交錯，而圖4-2(b)-(d)中的正方形影像跟圖4-2(a)一樣，不同在於圖4-2(b)-(d)中分別加入了50、100和150個雜訊資料點，我們便可利用對四種資料集合做偵測所產生的效率差異，來分析雜訊的影響力。此次實驗執行樣板理論的FCM、PCM、PFCM和IPCM每張圖片各100次，a和b值爲0.5，m值設爲2，n值設爲1.5 (此處m和n分別代表FCM和PCM的模糊化因子)， r_η 值限制爲0.9，初始給予三個叢集，叢集的初始大小以及位置皆爲隨機選取。表4-1爲圖4-2的實驗結果，表格中的數據爲做100次實驗平均所偵測到的叢集個數。以FCM的數據爲例，1.9代表做100次偵測平均一次可以偵測到1.9個叢集，然而此資料集合總共有3個叢集，因此最理想狀況每次都完全偵測，則實驗數據即爲3。

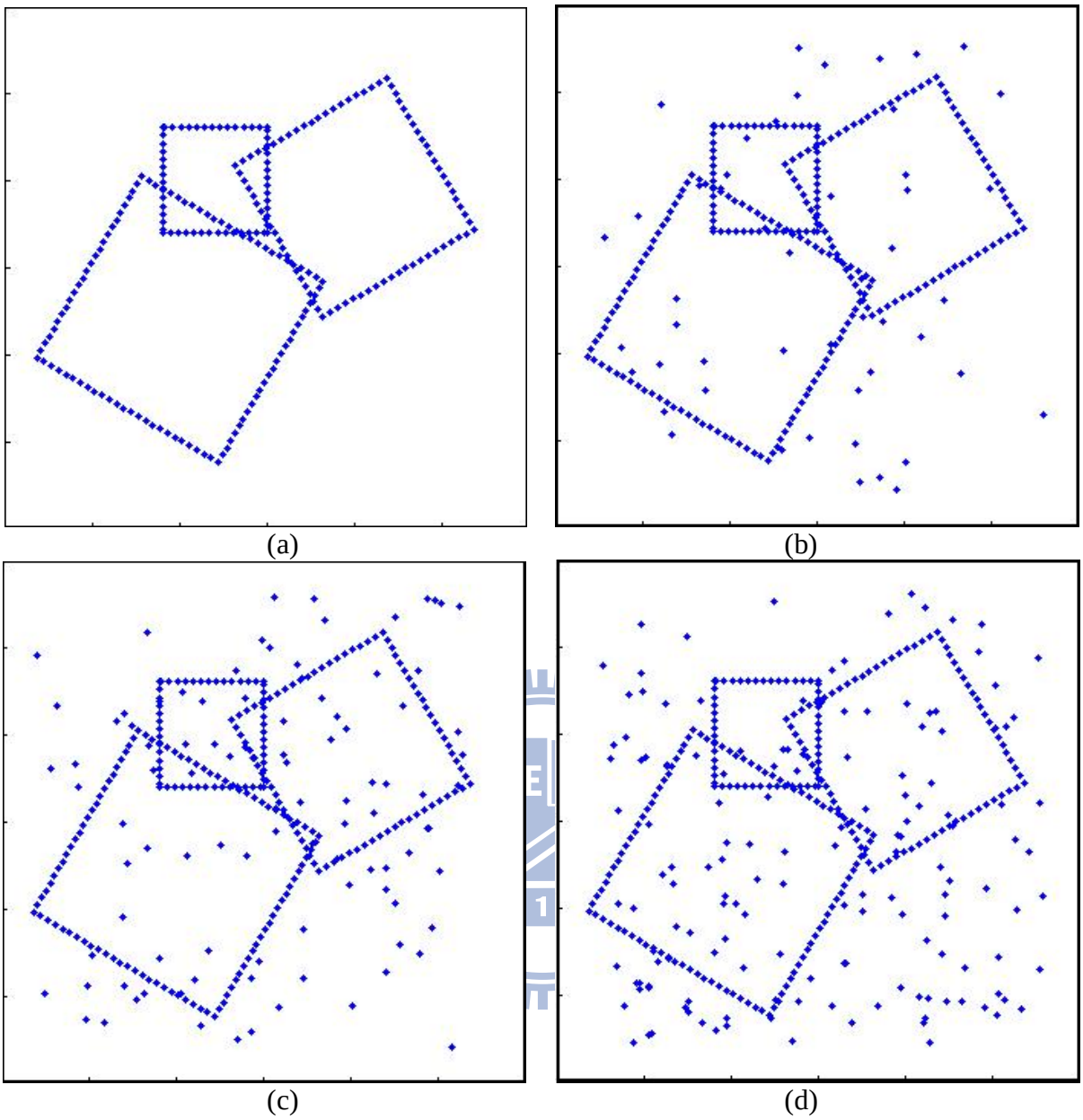


圖4-2：雜訊實驗圖 (a)原始影像 (b)加入50點雜訊資料 (c)加入100點雜訊資料 (d)加入150點雜訊資料

表4-1：雜訊實驗數據

雜訊個數	FCM	PCM	PFCM	IPCM
0	1.9	1.88	1.76	1.67
50	0.6	1.84	1.12	1.55
100	0.32	1.8	0.73	1.34
150	0.16	1.7	0.42	1.26

由實驗可知在加入雜訊資料前，四類演算法皆有相似的偵測效率。然而加入雜訊後，FCM明顯下降，PCM則沒有太大影響，然而PFCM和IPCM也有小幅度的降低。我們用圖4-3來做解釋，圖4-3(a)爲含有50個雜訊的影像，且影像中的叢集爲初始位置和大小，我們便以此設定來執行四種不同的演算法。圈起部分雜訊資料點的目的在於便於我們注意此群雜訊資料點對左上大正方形的影響力。

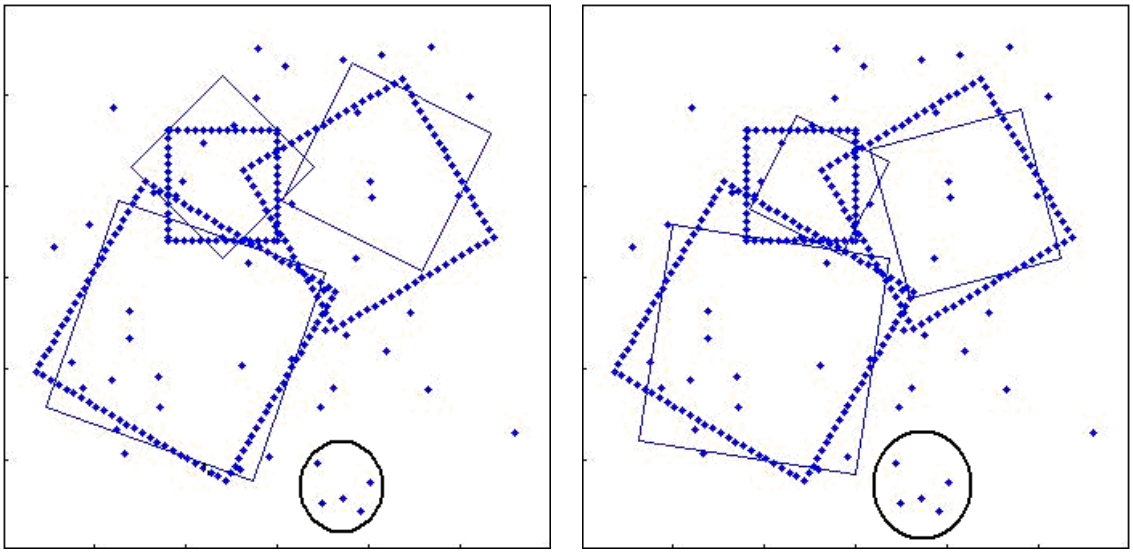
對於FCM，任意一點資料點的membership總和一定會等於1 (由(2))，因此就算此一資料點爲雜訊，它也可以擁有membership值來影響叢集的移動。如圖4-3(b)所示，這群雜訊資料所擁有的membership值平均有0.5左右，雖然數量不夠多，不至於會使整個叢集做大幅度的移動，但仍會影響叢集的偵測，使得偵測的效率變差。

而相對的，PCM沒有規定typicality總和一定會等於1，所以就算影像中有雜訊，可能雜訊點擁有的typicality值也會很小，不容易影響群中心移動，如圖4-2(c)，此群雜訊所擁有的typicality值平均0.01，叢集幾乎不受他們影響。

對於混合性叢集演算法而言，PFCM它的membership求法是先個別求出FCM和PCM的membership和typicality，然後再將兩者各取0.5比率相加，所得即爲PFCM的membership值，因此就數值涵義上來看，它不完全屬於FCM或者是PCM。當遇到雜訊時，FCM會受到雜訊影響，PCM可以使影響力降低，因此減少叢集受雜訊的影響度，如圖4-3(d)所示，這裡雜訊的membership和typicality值爲0.5和0.01，因此兩者混合後整體membership值差不多爲0.25，雖然仍有小幅度的影響，由表4-1顯示PFCM的效率比FCM來的好。

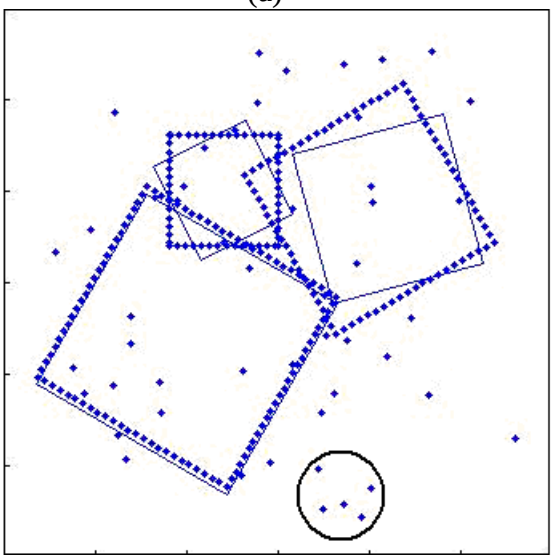
而IPCM的membership求法是FCM和PCM的membership值和typicality值相乘，由於兩值都是介於0到1之間，所以相乘的結果只會變的比FCM和PCM的membership和typicality值來得小而不會變大，因此會產生一種特性，只要其中一方判斷爲不是(membership值很小)，則IPCM演算法就會將其忽略，在此實驗中，因爲PCM對於雜訊

的影響力很小，因此就算FCM給予雜訊的membership值過高，在兩者相乘後，membership值仍會爲一個很小的數，如圖4-3(e)所示，這裡雜訊的membership和typicality值爲0.5和0.01，因此在兩者混合後雜訊整體的membership值平均0.005左右，雜訊的影響力大大的降低，這也就是爲甚麼在實驗數據中，IPCM的效率比PFCM來的要好。

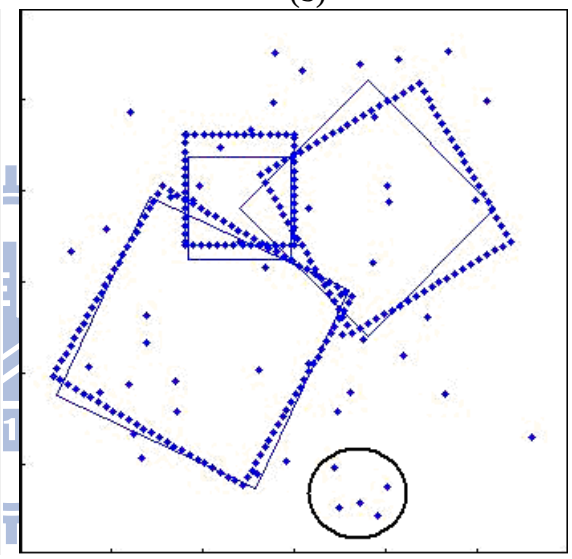


(a)

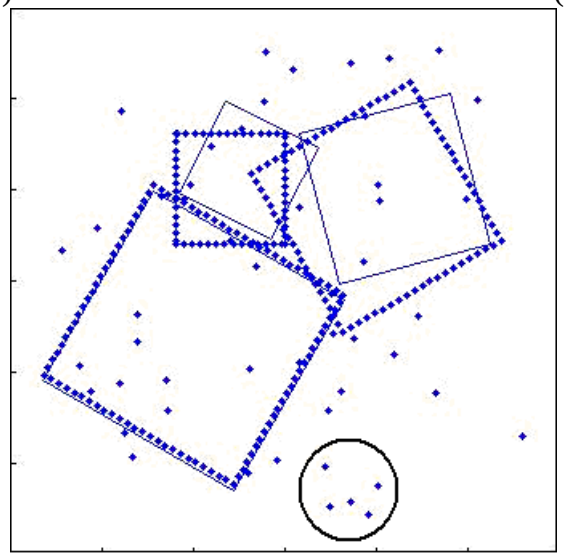
(b)



(c)



(d)

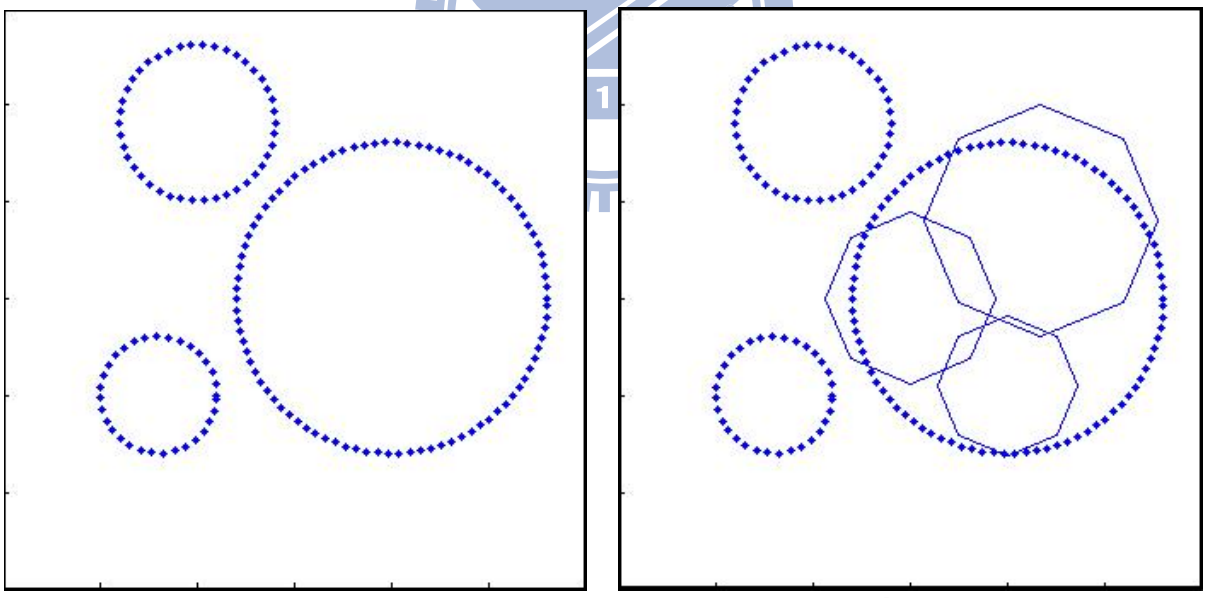


(e)

圖4-3：雜訊實驗示意圖 (a)爲初始的叢集位置，資料集合含有50個雜訊 (b)-(e)分別爲FCM、PCM、PFCM和IPCM執行的情形

4.2可能重複偵測的實驗

再來要探討的即是PCM所擁有的缺點，PCM演算法容易受叢集初始位置所影響，造成偵測的重複或者是錯誤。因此本次實驗的目的，即是測量混合性叢集演算法對此問題的改善效率。圖4-4(a)爲三個大小不同的圓形資料叢集，實驗中套入樣板理論的FCM、PCM、PFCM和IPCM演算法各執行100次，而演算法中的參數，a和b皆爲0.5，m值設爲2，n值設爲1.5，r_η值取0.9，初始給予三個叢集。而本次實驗的叢集初始位置，將限制放置於大圓處，如圖4-4(b)，而叢集的初始大小仍爲隨機，目的在於希望使得每個叢集都先注意到大圓資料集，增加重複偵測的可能性。表4-2爲此次實驗的實驗數據，數據所代表的涵義分別爲100次實驗中平均偵測到的大中小圓數量，而總和值即爲每次執行平均所偵測到的資料叢集個數。



(a) (b)

圖4-4：重複偵測的實驗 (a)原始影像 (b)初始叢集集中在大圓處

表4-2：重複偵測實驗數據

	FCM	PCM	PFCM	IPCM
大圓	0.98	1	0.99	0.98
中圓	0.6	0.06	0.32	0.48
小圓	0.33	0.01	0.12	0.29
總和	1.91	1.07	1.43	1.75

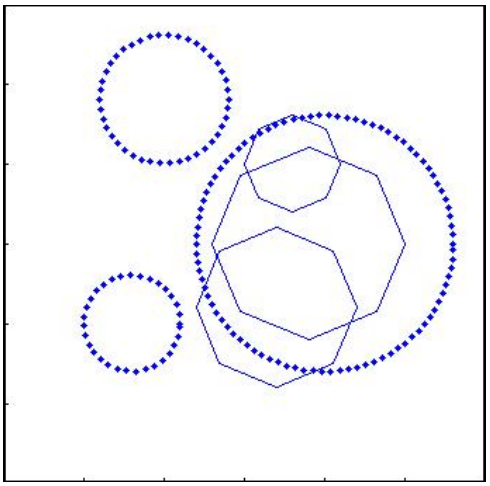
由數據中我們可以得知，因爲初始化的關係，大圓幾乎每次都會被偵測到，所以有無偵測到其他小圓，即爲比較的依據。我們以圖4-5來做輔助說明，圖4-5(a)爲一叢集初始圖，四種方法都以此初始圖來做執行，分別顯示執行的途中和結果的情形。

FCM的偵測效率爲四種方法中最好的，原因在於，FCM的membership總和一定會等於1 (由(2))，因此如果某一叢集搜尋到正確的位置時，則該叢集對於所涵蓋資料點的membership值將會趨近於1，而相對的其他叢集對於這些資料點的membership值將會很小(因爲相加要爲1)。因此在實驗中，如果某一叢集偵測到了大圓，則其他的叢集會因爲membership值太小而不會再去注意大圓而會往小圓方向移動並偵測，如圖4-5(b)(c)。這樣既不會偵測到重複的群而且叢集的中心位置也不容易受到初始位置所影響。

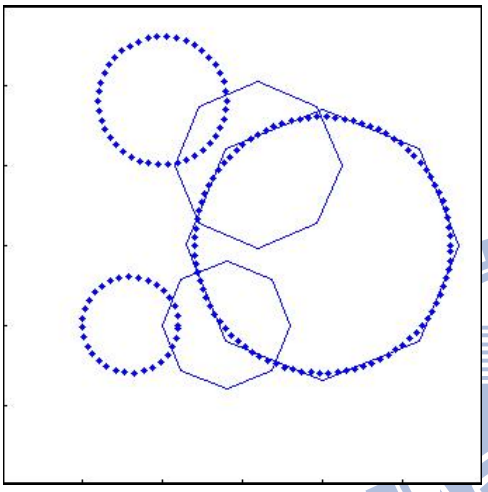
而PCM的效率就很差，幾乎每次都只能偵測到一個資料叢集(在這裡因爲實驗條件的設立，幾乎都是只偵測到大圓)，如圖4-5(d)(e)，原因在於membership總和不一定會等於1，叢集與叢集間缺少了溝通性，無法知道這個資料是否已經被其他的叢集所找到，因此可能會產生重疊偵測的事件發生。

PFCM由於混合了FCM和PCM兩種演算法特性，當PCM要產生重複的偵測時，FCM能夠有效的將兩者分開。但實驗數據中，PFCM的偵測效率並不像FCM一樣好，原因在於，雖然FCM能夠有效的將重複的叢集分開，但不代表他捨棄了typicality值，PCM仍有二分之一的影響力，因此就算叢集有機會去搜尋其他圓，仍會因爲大圓typicality值過高，叢集沒辦法完全脫離大圓，使得小圓的偵測不精確，如圖4-5(f)(g)。

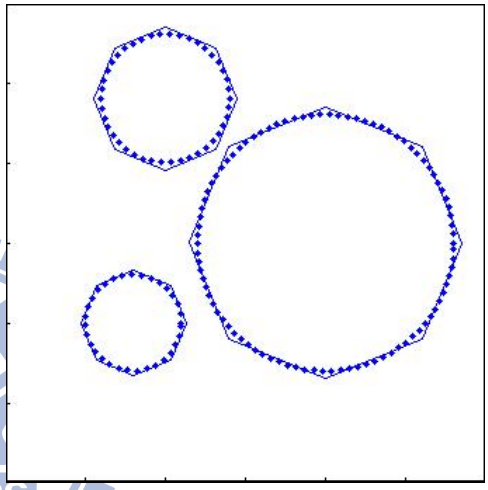
對於IPCM來說，它的membership求法是將個別求出FCM和PCM的membership和typicality值做相乘，不同於PFCM是各取一個比率相加，再加上membership和typicality的值都是介於0到1之間，這樣會使得只要其中一種演算法判斷爲不是(membership值很小)，則IPCM演算法就會將其忽略，因爲相乘後的membership值會變的更小。因此在此實驗中，當某一叢集找到正確的位置時，對於FCM來說，其他兩群對於這些被找到的資料點的membership值就會很小，就算PCM計算這些typicality值很大，兩者相乘仍會產生很小的數，使得重複偵測的可能性大大降低，可以順利偵測到其他小圓，如圖4-5(h)(i)，這也是爲甚麼數據中IPCM的偵測效率比PFCM來的高。



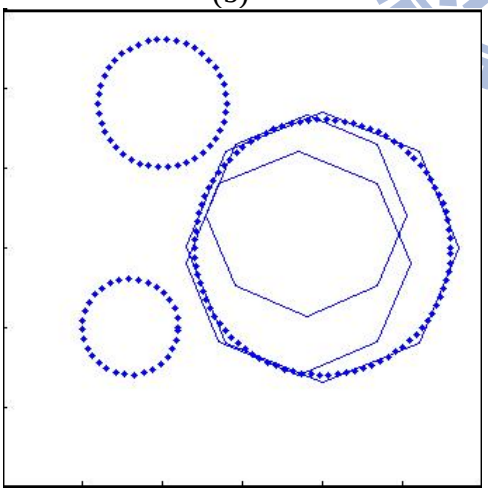
(a)



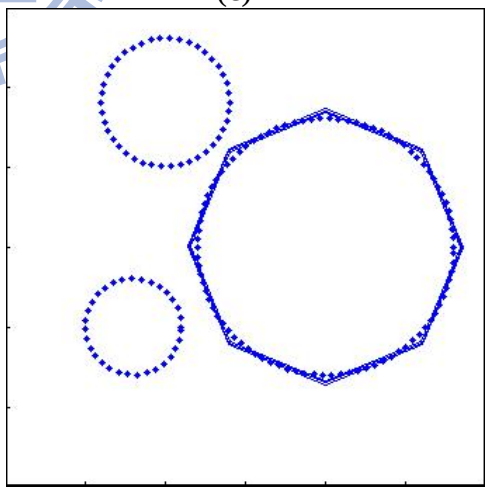
(b)



(c)



(d)



(e)

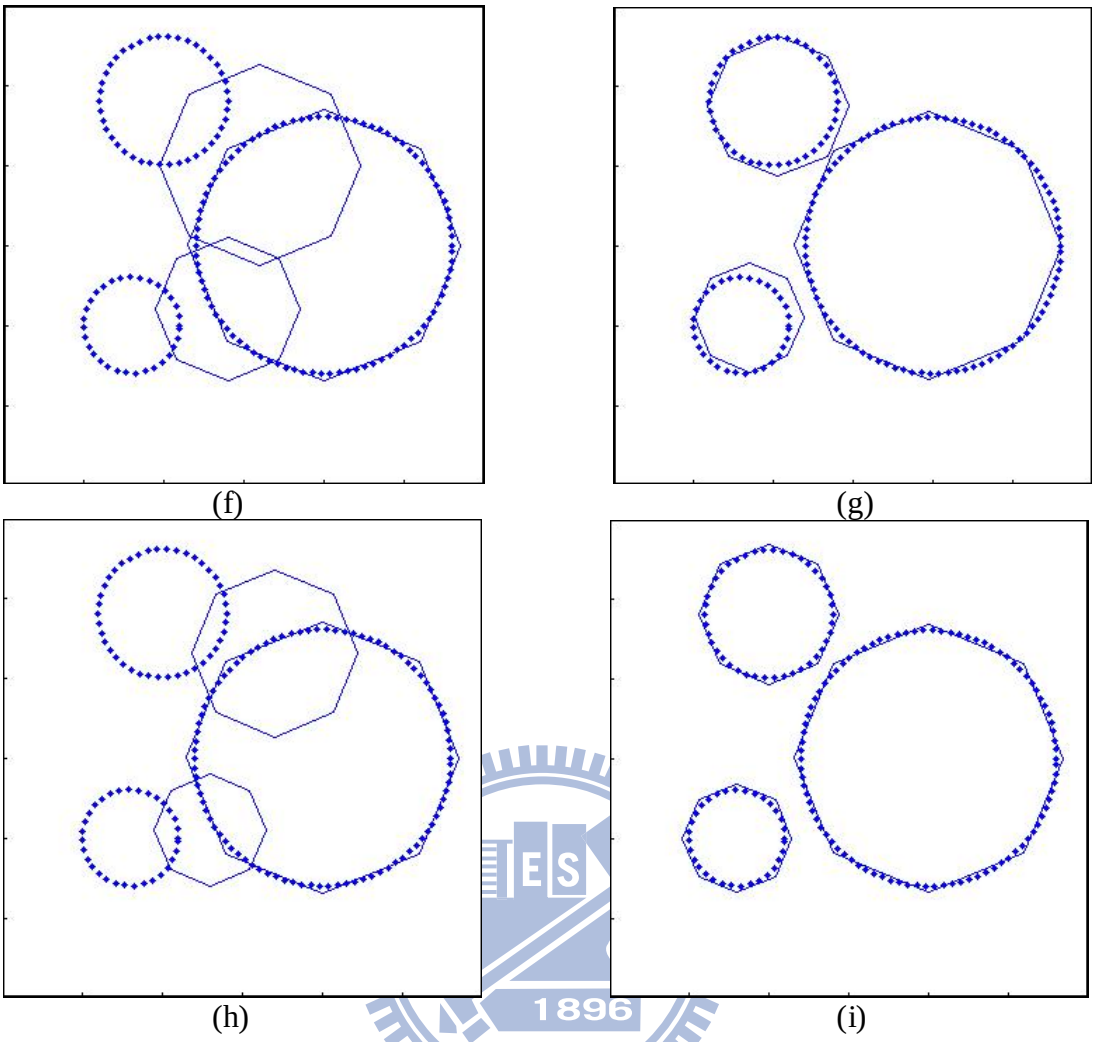
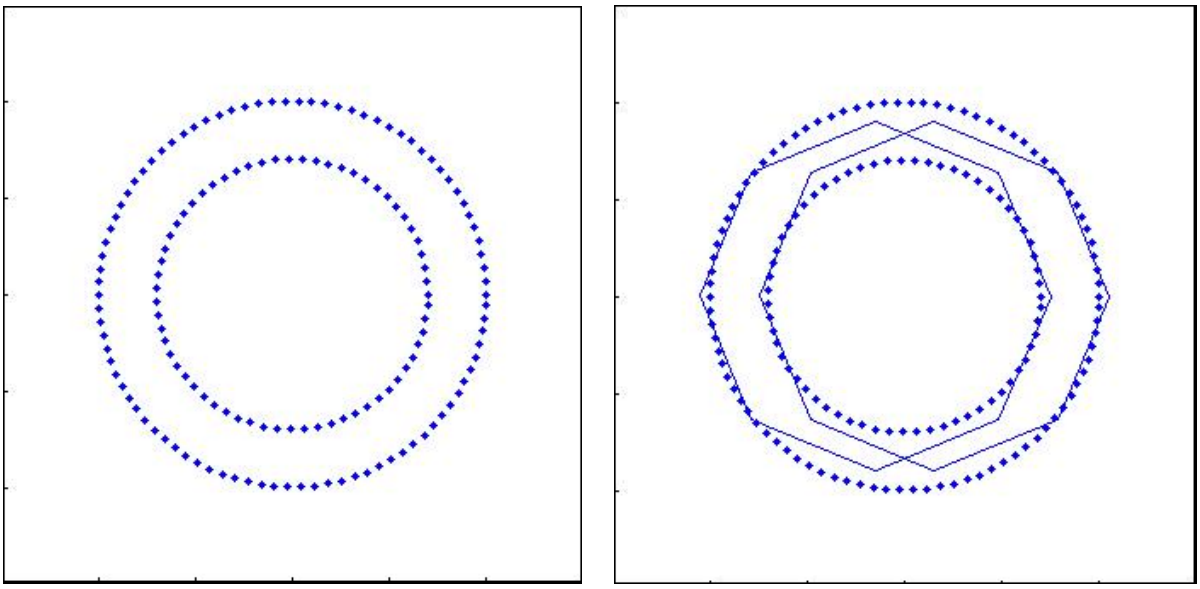


圖4-5：重複偵測實驗示意圖 (a)初始叢集位置 (b)(c)爲FCM執行途中和結果 (d)(e)爲PCM執行途中和結果 (f)(g)爲PFCM執行途中和結果 (h)(i)爲IPCM執行途中和結果

4.3 叢集個數與偵測效率

本章節的實驗目的，在於探討初始的叢集個數對偵測效率的影響力。這裡採用同心圓圖形(圖4-6(a))爲實驗的資料。我們知道在偵測同心圓圖形時，很容易偵測到錯誤的資料(圖4-6(b))，希望能藉由同心圓偵測，了解不同演算法在叢集個數的提升下，是否能擁有更好的效率。實驗所採取的參數，a和b皆爲0.5，m爲2，n爲1.5，r_η 值爲0.9，而初始的叢集位置以及大小皆爲隨機選取，每次實驗初始分別給予2、3、4、5個叢集，將FCM、PCM、PFCM和IPCM演算法各做100次做比較。表4-3即爲此次的實驗數，每個數據所代表的意義，皆爲每次偵測平均偵測到的正確叢集個數。



(a) (b)

圖4-6：叢集個數與效率實驗 (a)同心圓影像 (b)同心圓偵測容易產生的結果

表4-3：叢集個數與效率實驗數據

初始叢集個數	FCM	PCM	PFCM	IPCM
2	0.54	0.92	0.75	0.34
3	0.48	1	1.67	0.32
4	0.38	1	1.64	0.28
5	0.43	1	1.6	0.36

由實驗結果我們可以看到當初始給予兩個叢集時，FCM、PFCM和IPCM的效率很差，幾乎都會產生圖4-6(b)的錯誤偵測，而PCM的偵測效率趨近於1，原因是在於η值的收斂速率 r_η 較慢，因此演算法偵測的範圍不容易變小，比較容易留意並且偵測到外圓，但因爲叢集會產生重複，幾乎都是偵測到外圓而非內圓。

如果慢慢增加初始給予的叢集個數，我們可以發現，FCM和IPCM效果仍然很差，多餘的叢集只會徒增辨識上的困難，叢集會分散資料點的membership使他們的偵測更爲混亂，如圖4-7(a)(b)(c)。3張圖分別將不同叢集的偵測情形顯示出來，叢集爲同時給予。而PCM的偵測效率稍爲提升，原因在於PCM容易受初始的叢集位置所影響，因此如果

給予的叢集個數愈多，所能正確偵測到的機率也會愈高。但因此時並未更改 r_η 值，因此所能偵測到的仍然爲外圓，如圖4-7(d)(e)(f)，所以效率仍趨近於1。

重點在於PFCM，只是增加了一個叢集個數卻使得偵測效率大大提升，主要的原因在於，PFCM不會完全採用FCM或PCM的membership或typicality來做偵測，就算叢集偵測錯誤，所被取走的membership也只僅限於FCM的，所以這些資料點仍有機會被下一個叢集偵測到。就此次實驗來說前兩個叢集可能會產生圖4-6(b)的錯誤，然而內圓資料點因爲錯誤的偵測FCM的membership值已經消失，雖然仍含有PCM的typicality，但其影響力已經不比外圓來的大，所以第三個叢集才比較有機會偵測到外圓，如圖4-7(g)(h)(i)。因此，如果前兩個叢集偵測錯誤，第三個叢集便有很大的機率能偵測到正確的資料，使得每次執行基本上都會偵測到一個正確叢集，但也可能同時偵測到兩個叢集，所以由表4-3偵測效率會大於1。

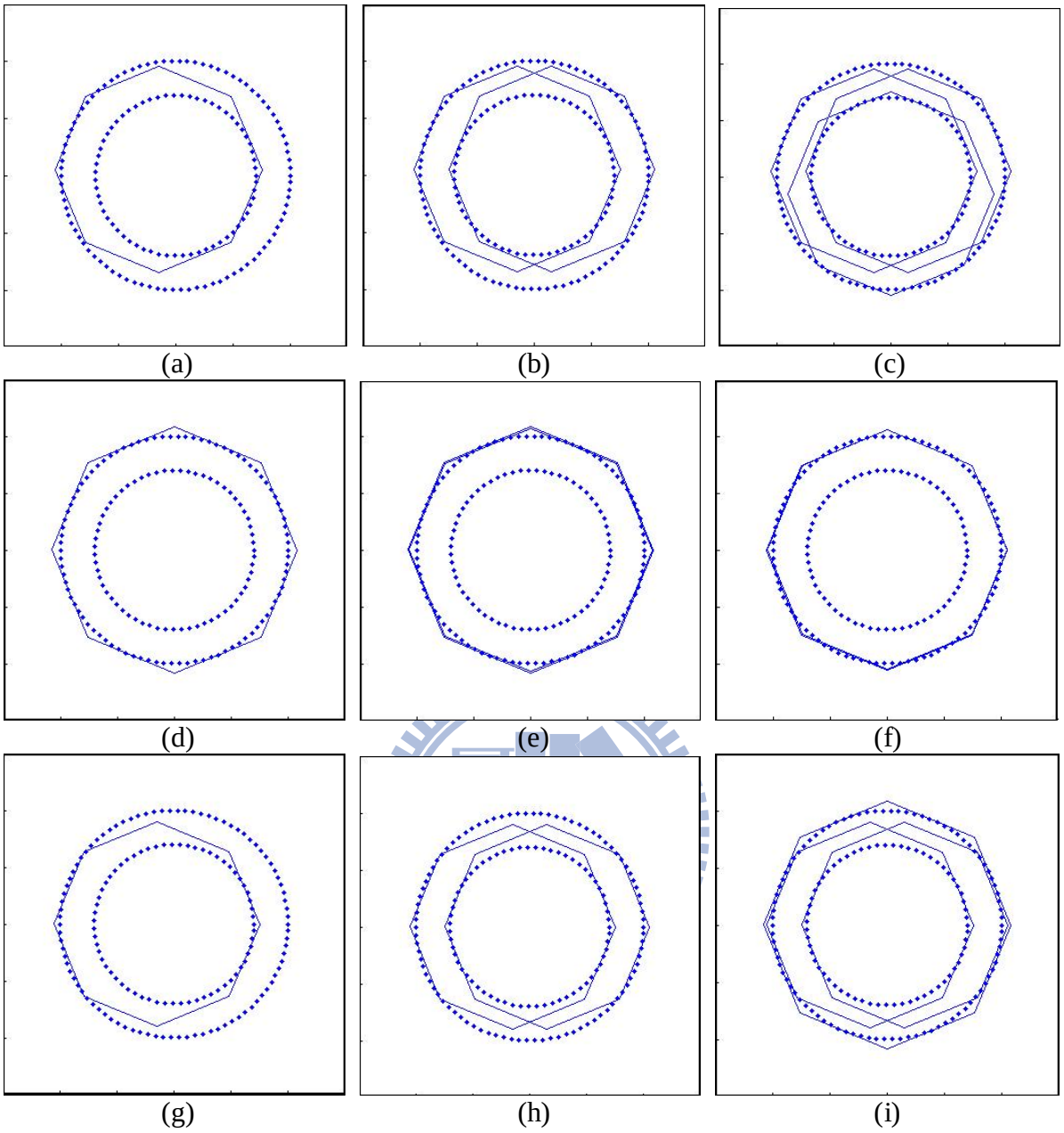


圖4-7：同心圓偵測解析 (a)-(c)依序爲FCM和IPCM三個叢集的分群結果 (d)-(f)依序爲PCM三個叢集的分群結果 (g)-(i)依序爲PFCM三個叢集的分群結果

舉個圖例來解釋，假設圖4-8(a)爲兩個交錯圖形容易產生的偵測錯誤。以FCM來說，當某一叢集已經視某資料點爲同一叢集，則其他叢集是不可能將此資料點也視爲同一叢集。這樣的問題在於，如果其中有一叢集偵測錯誤，則後面的叢集要能偵測出正確的資料會變得非常不容易，如圖4-8(b)，當有一個叢集偵測到錯誤的圖形後，會使得這些被

偵測錯誤的資料點所屬的membership值都趨近於1，其他的叢集對於這些資料點便不會注意(membership趨近於0)，有如這些資料點不存在一般，這樣剩下的叢集要找到正確的正方形就很困難。

就PCM而言，可能1個資料點對於兩個叢集的typicality值都會等於1，這樣的特色在於，某一叢集偵測到資料點，並且將此歸納爲同一叢集，其他叢集也可能視此資料點爲同一叢集，因此會產生重複的情形發生，所以，如果資料集容易產生某一錯誤偵測，則PCM所做出來的結果可能會全部都偵測錯誤，這樣不管初始給予再多的叢集也無法完成。如圖4-8(c)，不管叢集數量給得再多，也會偵測到錯誤的圖形。

而 PFCM 不會完全採用 membership 或 typicality 來做偵測，就算叢集偵測錯誤，所被取走的 membership 也只僅限於 FCM 的 membership，所以這些資料點仍有機會被下一個叢集偵測到，然而對於容易偵測錯誤的資料來說，整體的 membership 少了 FCM 的 membership 這一部分，之後也很難有機會再偵測到相同的錯誤。如圖 4-9，由於外圍的 membership 已經被取走一半，所以不容易偵測到相同的錯誤，剩下的叢集比較有機會往裡面走。

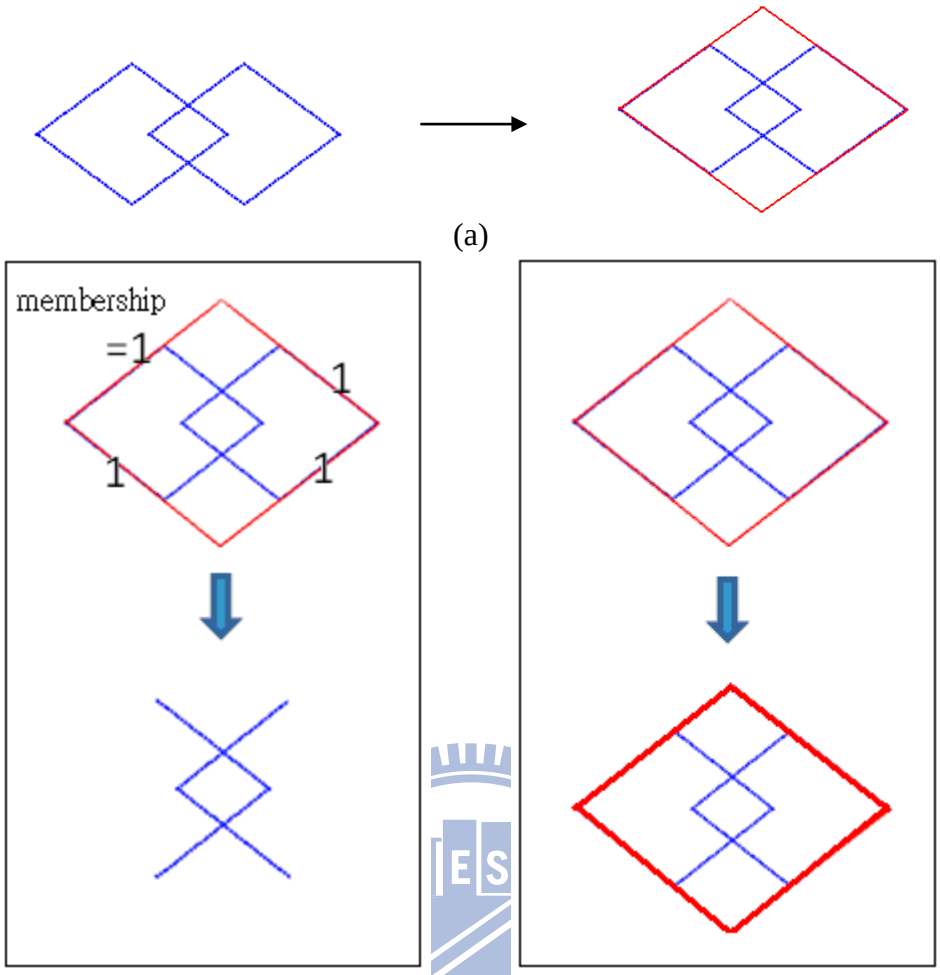


圖4-8：FCM和PCM會產生問題的示意圖 (a)叢集偵測可能出現的錯誤 (b)FCM偵測過後因爲資料點membership趨近於1，對其他叢集來說被偵測的資料點有如不存在一般 (c)PCM可能會一再偵測到重複的錯誤

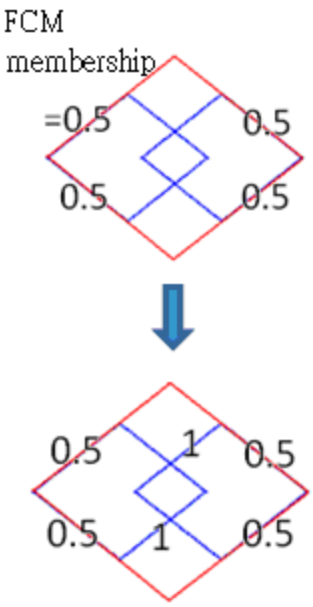


圖 4-9：PFCM 的偵測示意圖，上圖數字代表整體的 membership 只被取走 FCM 的 membership 這一部份，下圖數字代表仍剩下的 membership

圖4-10爲一些實際操作的例子。圖4-10中分別顯示當FCM、PCM和PFCM的第一個叢集偵測錯誤後，第二個叢集的偵測情形，我們可以注意到FCM和PCM都沒有辦法找到正確的叢集，而PFCM則可以找到一個正確的叢集。

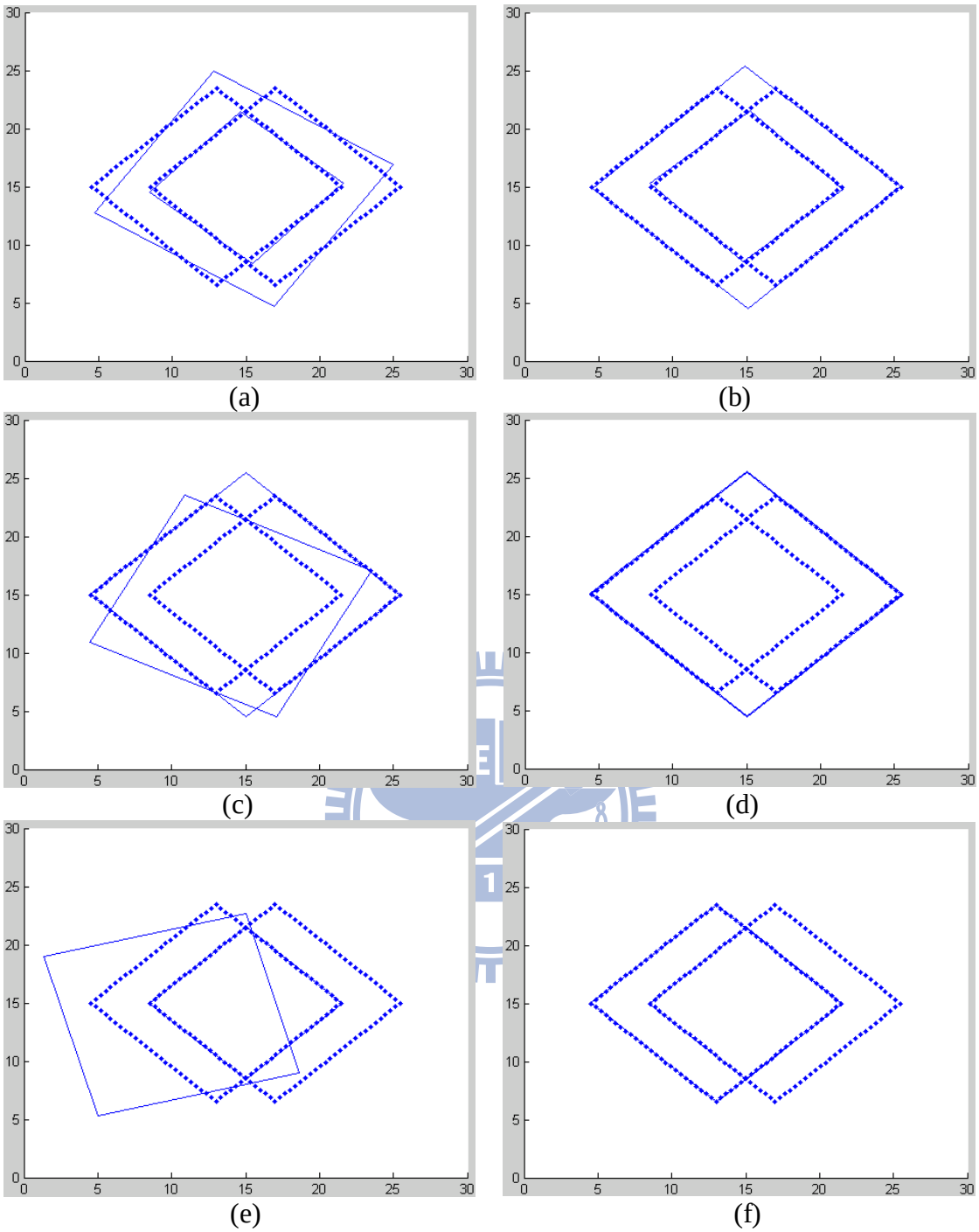


圖4-10：偵測實際操作 (a)FCM第一個叢集偵測錯誤 (b)FCM第二個也偵測錯誤 (c)PCM第一個叢集偵測錯誤 (d)PCM第二個也偵測到同一個錯誤點上 (e)PFCM第一個叢集偵測錯誤(f)PFCM第二個叢集偵測正確

由前面的理論我們知道FCM所出現的問題，當有一個叢集偵測到錯誤的資料後，會使得這些被偵測錯誤的資料點所屬的membership值都趨近於1，其他的叢集對於這些資料點的membership會趨近於0，使得資料點有如消失一般。PFCM就是由此下手，雖然取走了FCM的membership，但還是可以藉由PCM的typicality來做偵測，重點在於PFCM的membership值是兩者的相加，FCM不會影響到PCM的數值。但是IPCM並不像PFCM一樣是相加制，它是將兩個值相乘，所以如果FCM的membership趨近於0，則相乘後整個membership也會趨近於0，這樣會使得IPCM產生跟FCM一樣的問題，因此IPCM沒有PFCM這樣的特性，這也是爲什麼在以上的實驗數據中，IPCM的效率會那麼的差。

再來我們便針對幾個複雜的圖形做測試，如圖4-11(a)-(d)，來看看混合性叢集演算法的偵測效率，所採用的各類參數跟之前一樣，針對FCM、PCM、PFCM和IPCM做實驗分析，而實驗數據分別爲表4-4。由實驗結果可以看到PFCM和IPCM大多都比FCM和PCM來的更好。

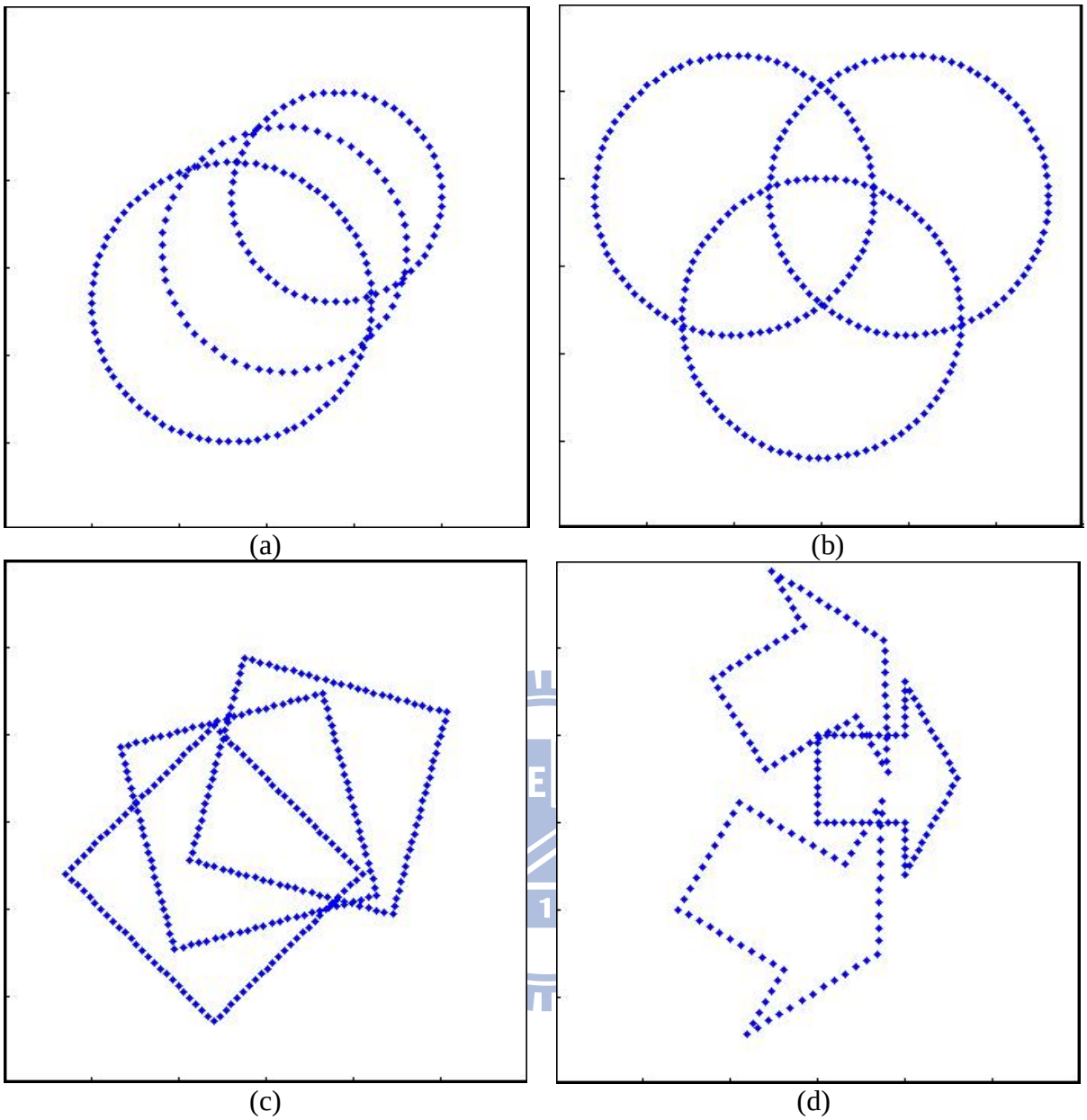


圖4-11：幾個複雜的圖形 (a)(b)三圓交叉 (c)三個正方形交錯 (d)三個箭頭

表4-4：複雜圖形的偵測數據

圖形編號	FCM	PCM	PFCM	IPCM
(a)	1.84	1.92	2.47	2.42
(b)	2.5	2.08	2.8	2.16
(c)	1.56	1.71	2.62	2.35
(d)	1.16	0.96	1.21	1.12

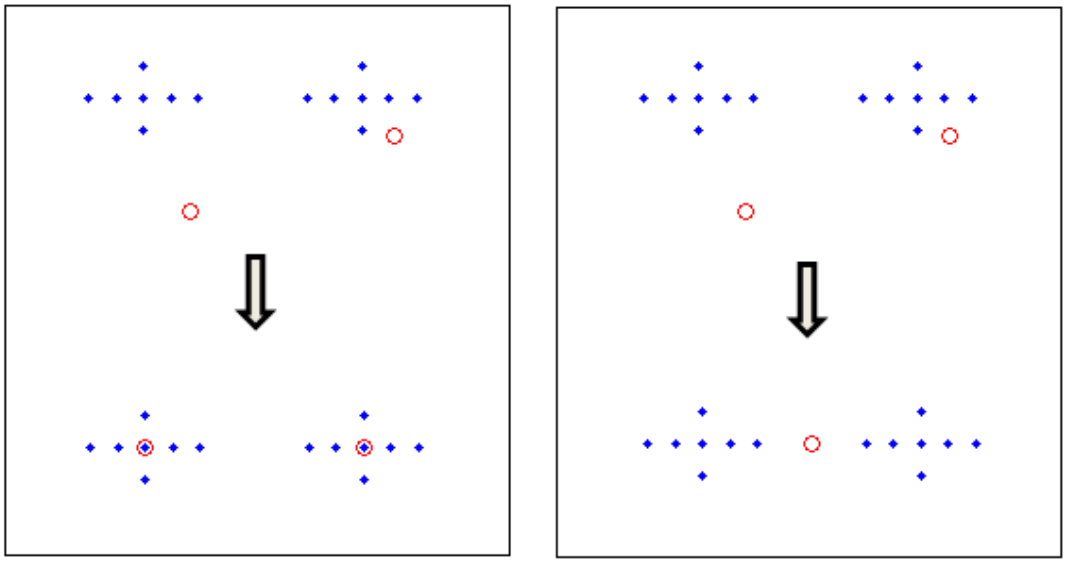
4.4 η值的分析

在前面的章節，我們皆是固定 r_η 值等於0.9來做實驗分析，因此可能會錯過一些關於η值所產生的影響，因此本章將針對η值做分析討論，藉由改變不同的 r_η 值，來產生不同的收斂速率。在做實驗前，我們先來了解一下η值的涵義

我們分別來討論當η值收斂太慢或者太快時，會產生甚麼樣的問題。此時所探討的內容，僅限於PCM演算法會產生的問題。

I. r_η 值趨近於1(η值收斂太慢)：

由方程式(6)中可以知道當η值太大時，整體typicality值會趨近於1，因此叢集會聚集在所有資料點的中心位置。如圖4-12，(a)爲η值正常收斂時(r_η 值=0.8)，叢集會分別偵測到兩群的中心位置(b)爲η值收斂太慢時(r_η 值=0.99)，叢集則會視所有的資料爲同一群

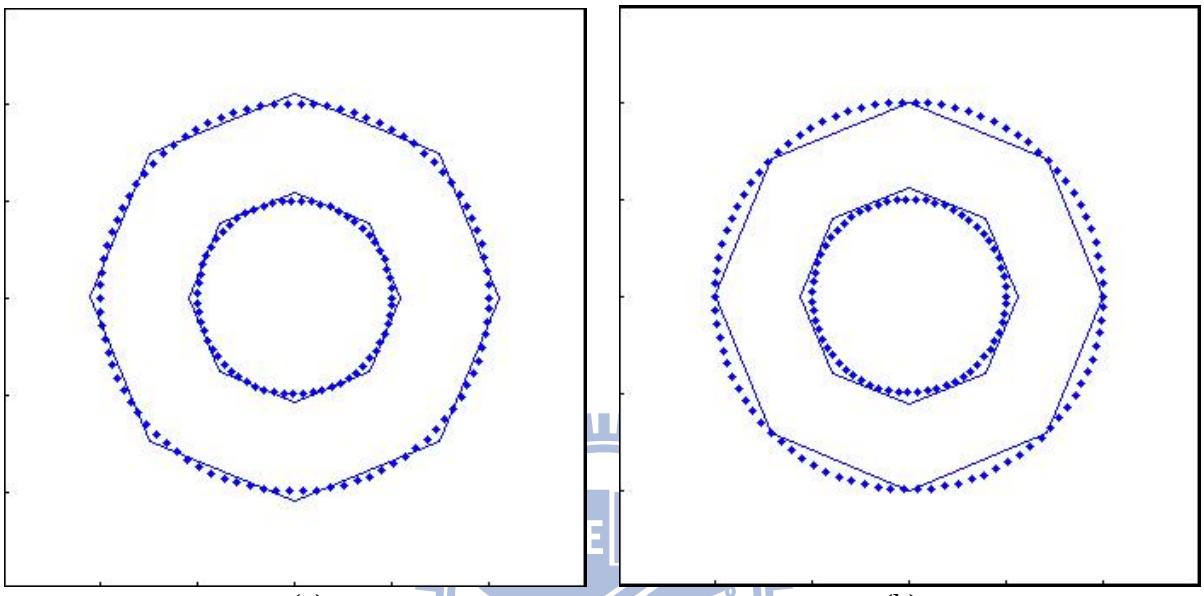


(a) (b)

圖4-12：η值收斂太慢所產生的問題一 (a) r_η =0.8 (b) r_η =0.99

然而，資料點與叢集的距離如果很近，當η很大時，typicality值的計算值會差不多，

這樣會造成叢集無法移動到精確的位置，使得辨識有些許誤差。如圖4-13，(a)爲η值正常時(r_η 值=0.7)，(b)爲η值收斂太慢時(r_η 值=0.99)，可以發現兩著的叢集精準度有些微的差距。



(a) (b)

圖4-13：η值收斂太慢所產生的問題二 (a) r_η =0.7 (b) r_η =0.99

II. r_η 值趨近於零(η值收斂太快)：

由方程式(6)可以知道typicality會因爲η值太小而造成整體偏低的現象，而由方程式(4)可以得知typicality太小會使得資料的存在性過低，叢集不容易移動到正確的位置上，如圖4-14。

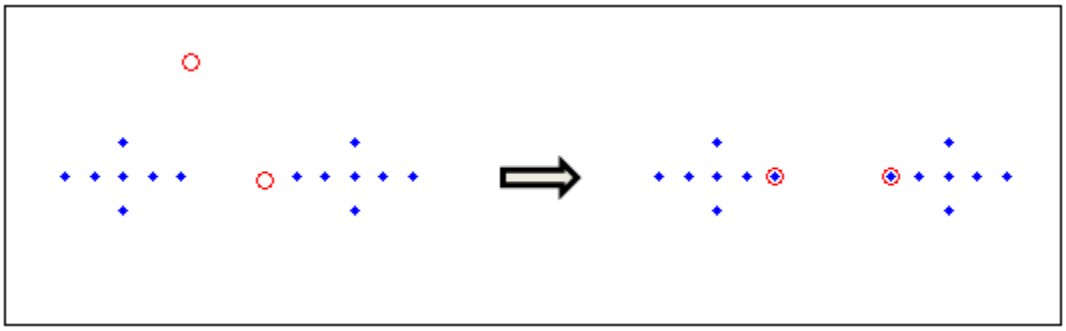
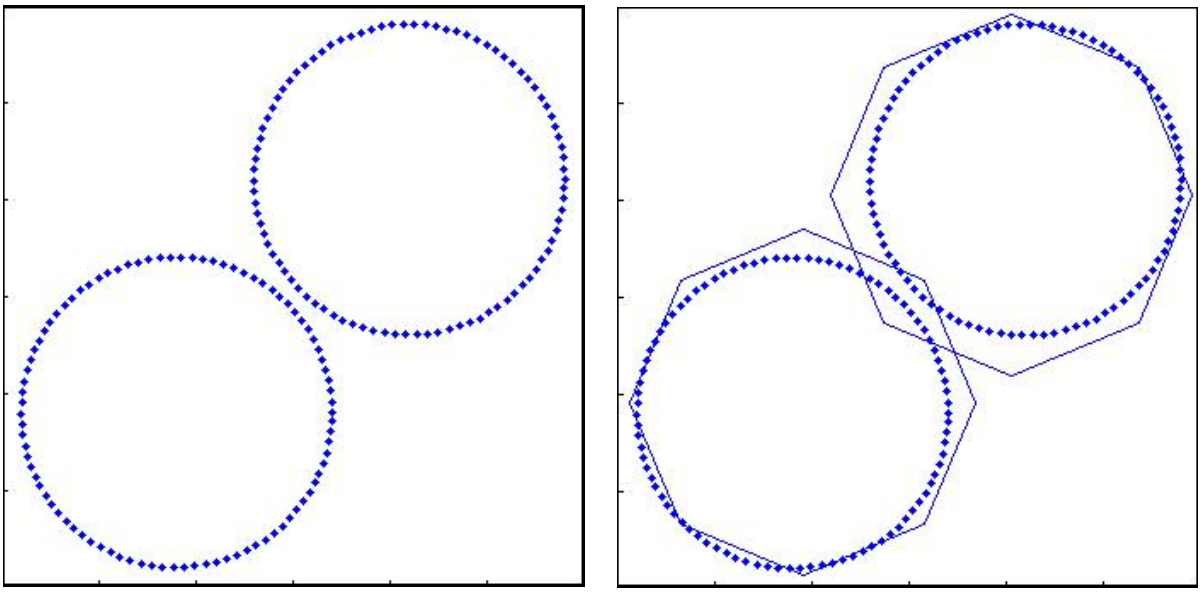


圖4-14：η值收斂太快時叢集無法移動到正確的位置

由上面的討論，我們可以知道η值的重要性，然而這只是單純用PCM的觀點去探討而已，之後便要在混合型的叢集偵測演算法上做實驗分析，看有甚麼樣子的結果。

由於混合性叢集演算法是結合FCM和PCM的一種新的理念，不能單以PCM去解釋η值所造成的問題，因此本次實驗目的在於探討η值對於混合性叢集演算法的影響。此次實驗我們以兩個相近的圓形做爲實驗資料集，如圖4-15(a)。所引用的參數，a和b皆爲0.5，m爲2，n爲1.5，r_η 值依序給予0.99、0.9、0.8、0.5、0.2和0.1，初始給予2個叢集，叢集的初始位置和大小皆隨機選取，對FCM、PCM、PFCM和IPCM各做100次實驗分析。表4-5爲此次的實驗數據，數據的涵義爲每次偵測到正確的叢集個數。



(a) (b)

圖4-15：η值分析實驗 (a)實驗初始圖 (b)當η值收斂太慢時PFCM容易產生的錯誤

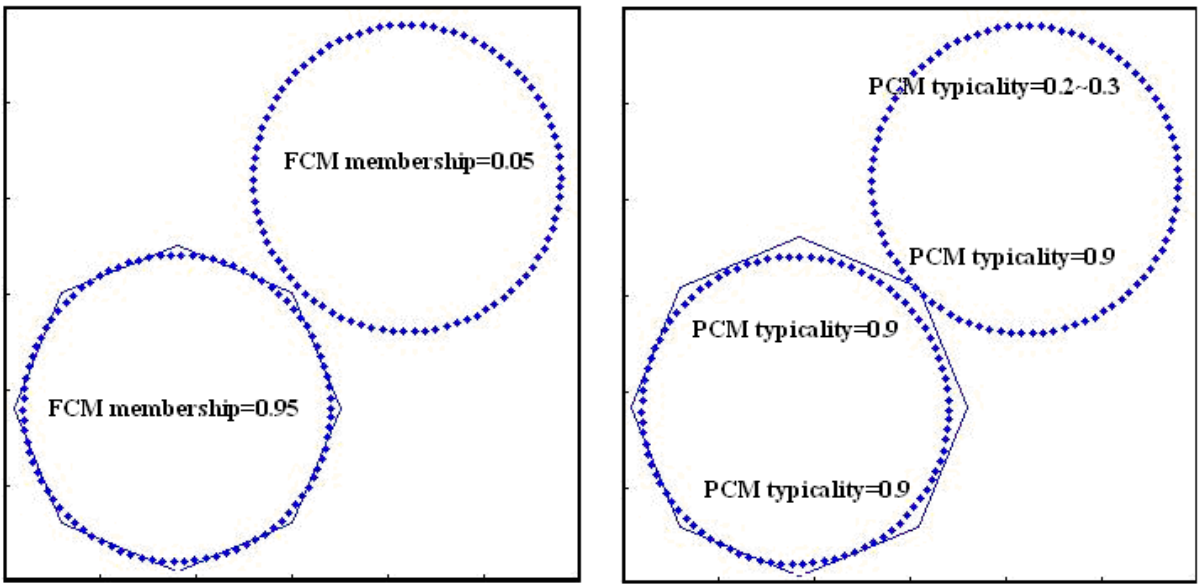
表4-5：η值分析實驗數據

r_η	FCM	PCM	PFCM	IPCM
0.99	1.76	1.15	0.3	1.45
0.9	1.76	1.23	1.54	1.25
0.8	1.76	1.23	1.5	1.16
0.5	1.76	1.08	1.5	1
0.2	1.76	1.07	1.51	0.97
0.1	1.76	1.06	1.52	0.96

由數據中我們可以看到，FCM本身沒有η值參數，故不會受影響，PCM也沒有太大的變動，PFCM的 r_η 值0.99和0.9時偵測效率差非常多，因爲 r_η 值0.99時幾乎都會出現如圖4-15(b)的錯誤偵測，原因在於前一小節中我們知道當時 r_η 值趨近於1時，會造成叢集容易移動到所有資料點的中心，而且會有測量不精準的問題，所以容易出現誤差的情形。這樣的問題在PFCM中也會產生，叢集容易卡在所有資料的中間，雖然表面上來看

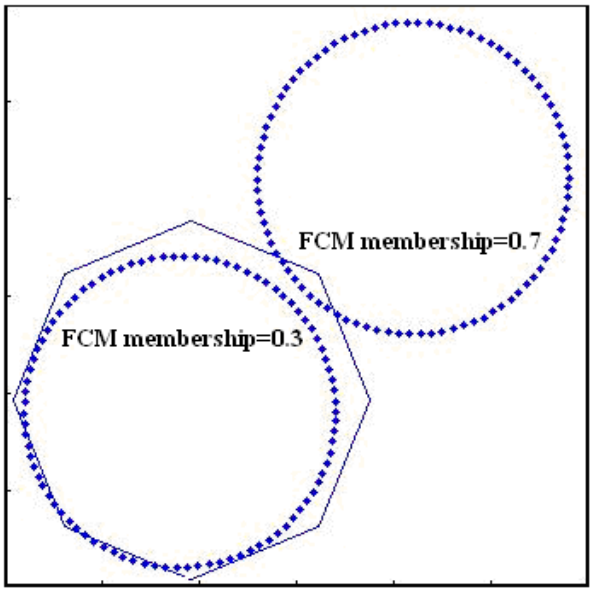
會感覺PCM的typicality所占比率只有一部分，還有FCM的membership可以加以修正，但這樣的想法是不嚴謹的，因爲FCM可能會將叢集拉到其他的位置去，造成更不精準的情形，如圖4-16。

圖中我們可以看到，PFCM中的FCM因子在計算後可能已經判定了叢集的正確位置，如圖4-16(a)偵測到的左下圓形的membership爲0.95，但是由於η值太大，因此在PCM的計算中，認爲此叢集應該是要往中間的位置移動，如圖4-16(b)可以注意到中間資料點的typicality值皆爲0.9，這時FCM membership很容易就會被其他的資料所影響，使得整個叢集被拉到另一圓圈去，如圖4-16(c)FCM反而認爲另一邊的圓才是所要偵測的，而使得整體叢集往另一圓移動。這也就是爲甚麼當 r_η 值爲0.99時，PFCM的效率會這麼差。而就IPCM而言，當 r_η 值爲0.99時是偵測效率最好的時候，原因在於，η值過大只會使得PCM的typicality較容易趨近於1，membership和typicality兩者相乘的結果，就會趨近於FCM的membership，因此IPCM會趨近於FCM演算法，在此次實驗中FCM擁有最高的偵測效率，這也是爲甚麼IPCM在 r_η 值愈趨近於1時，偵測效率會愈好。



(a)

(b)



(c)

圖4-16：η值太大時PFCM流程圖 (a)叢集偵測到了圓 (b)因爲η值太大所以叢集會跑到所有資料的中心位置 (c)叢集被另一邊的圓拉過去

再來我們以圖4-17做爲資料集，針對FCM、PCM、PFCM和IPCM的偵測做分析，圖中三個叢集較爲分散，因此對於PCM來說，η值的選取會變得相當重要，不然不容易搜尋到較遠的叢集，容易產生重疊。初始給予三個叢集做分析，參數都與前者相同，而所採取的 r_η 值分別爲：0.99、0.9、0.8、0.5、0.2和0.1。表4-6即爲此次的實驗數據。

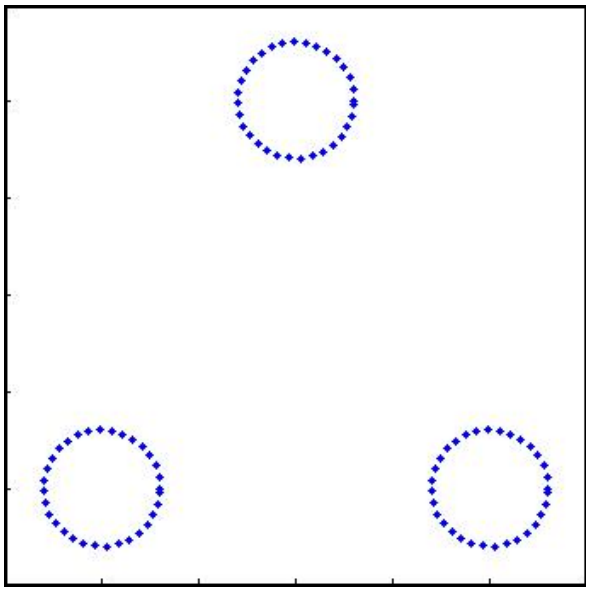
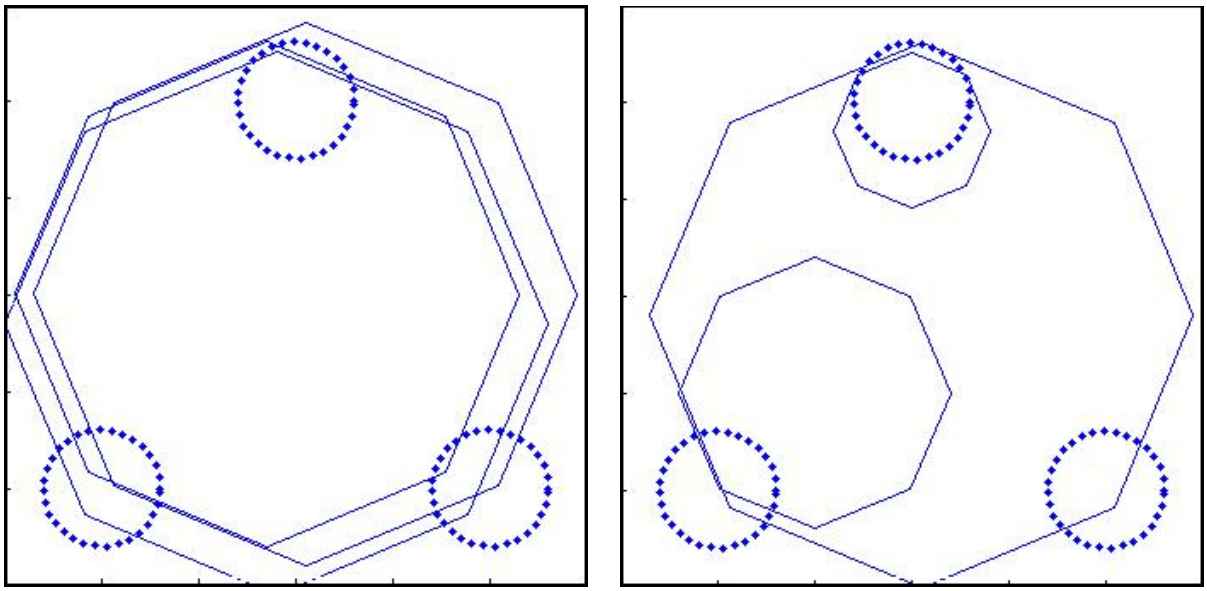


圖4-17：η值分析實驗2初始圖，三個分散的小圓

表4-6：η值分析實驗2

r_η	FCM	PCM	PFCM	IPCM
0.99	1.5	0.4	0.5	1.11
0.9	1.5	0.72	0.72	0.5
0.8	1.5	0.62	0.74	0.32
0.5	1.5	0.38	0.92	0.24
0.2	1.5	0.36	1.15	0.19
0.1	1.5	0.36	1.18	0.2

由前一次實驗我們可以了解，當 r_η 值趨近於1時，PFCM和IPCM數據的產生原因。PFCM所有的資料點會被視爲同一群，使得叢集被拉到所有資料點的中心位置，如圖4-18(a)。而IPCM則會趨近於FCM，擁有不錯的效率。而當 r_η 值慢慢變小以至於趨近於0時，IPCM的效率迅速下降，因爲 r_η 值趨近於0的話，會使PCM的typicality值也趨近於0，membership和typicality相乘後會連帶的使整體的membership都趨近於0，叢集不容易移動到正確的位置上，如圖4-18(b)。而對於PFCM來說，η值如果太小的話，PCM的typicality值趨近於0，所以PFCM只會被剩下FCM的membership所影響，所以也會比較傾向於FCM，這也是爲甚麼在此次實驗中，當PFCM在 r_η 值愈趨近於0時，偵測效率會較好。



(a) (b)

圖4-18：η值分析實驗結果圖 (a) PFCM當η值太大時所有的資料點會被視爲同一群，使得叢集會被拉到所有資料點的中心位置 (b)IPCM當η值太小時，叢集不易移動到正確的位置

第五章：結論

叢集演算法可以幫助我們從未知的資料中，找出資料分佈的狀況與模式，其目的是在分析資料的內容，將性質相似的資料連繫在一起。然而對於shell cluster偵測的困難度也會上升，雖然可利用較傳統的FCM和PCM演算法做搜尋，但彼此仍存在著一些限制，無法單一的使用一種演算法做廣泛的運用。

在去雜訊以及減少重複性的研究中，我們可以發現到，PFCM的偵測機率皆位於FCM和PCM的中間，那是因爲PFCM的設計概念是將membership和typicality各自取一比率相加，因此效果不會來的比某一個方法好，但也不會比另一個方法差。這樣雖然能同時解決一定的雜訊和重複問題，但是由之前的實驗可以得知，IPCM在去雜訊和解決重複偵測的問題，都比PFCM來的更好。

然而在初始叢集個數不同的實驗中，我們發現到PFCM有著降低錯誤率產生的特性，這樣的好處在於，在做資料分群時，由於一次就分好所有群的機率很低，因此會採用逐一解決的方法，先偵測到的叢集便將它從資料集中移走(remove)，然後再用剩下的資料集做偵測，直到所有的叢集都找到爲止。因此如果是採用FCM或者是PCM的演算法，可能一個正確的資料叢集都沒有辦法找到，然而PFCM就會比FCM和PCM要來的更有機會，準確率也會比較好。尤其是針對shell cluster做偵測時，如果圖形過於複雜，很容易會造成偵測上的失誤，因此在演算法的選擇上就成了辨識上的關鍵，這時PFCM就會是個不錯的選擇。

在實驗的最後我們討論到η值會影響偵測的範圍，如果η值收斂太爲緩慢，容易使得typicality值趨近於1，可能會視所有資料爲同一群而偵測到所有資料的中心位置，但對於IPCM來說，不會產生這樣的錯誤。而當收斂太迅速，加上初始雛形位置不夠廣時，

會使得typicality值趨近於0，使得叢集不容易移動至正確的位置造成偵測的失誤，而PFCM則可以改進這樣的問題。

藉由本篇論文的分析，讓我們瞭解到了PFCM和IPCM的不同性質，使我們在偵測較複雜的叢集時，有著更高的效率。在未來的研究上，可以針對允許初始雛型的扭曲或變形做分析，也可以在初始一次給予多種圖形形狀做偵測，藉此近一步分析混何性叢集演算法的特性，希望在未來PFCM和IPCM這兩種混合性叢集演算法，能帶來更大的使用效益與使用空間。

參考文獻：

[1] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, New York: John Wiley & Sons, 1973.

[2] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, N.J.: Prentice Hall, 1988.

[3] R.N.Dave and S.Bhamidipati, "Application of fuzzy shell-clustering algorithm to recognize circular shapes in digital images", *Proc. Third Annual IFSA World Congress*, pp. 238-241, 1989.

[4] R.N. Dave, "Fuzzy shell-clustering and application to circle detection in digital images", *Int'l J. Gen. Syst.*, vol. 16, pp. 343-355, 1990.

[5] R. N. Dave and K. J. Patel, "Fuzzy ellipsodial-shell clustering algorithm and detection of ellipsoidal shapes," *Proc. SPIE Conf: Intelligent Robots and Computer Esion IX: Algorithms and Techniques*, vol. 1381, pp. 320-333, 1990

[6] R.N. Dave and K. Bhaswan, "Adaptive fuzzy C-shells clustering and detection of ellipses", *IEEE Trans. Neural Networks*, vol. 3, pp. 643-662, 1992.

[7] F. Hoeppner, "Fuzzy shell clustering algorithms in image processing: fuzzy c-rectangular and 2-rectangular shells", *IEEE Trans. Fuzzy Systems*, vol. 5, pp. 599-613, 1997.

[8] X.-B. Gao,W.-X. Xie, J.-Z. Liu, and J. Li, "Template based fuzzy c-shells clustering algorithm and its fast implementation", *Proc. IEEE Int'l Conf. Signal Processing*, pp. 1269-1272, 1996.

[9] Wang, T, "Possibilistic Shell Clustering of Template-Based Shapes", *IEEE Trans. Fuzzy Systems, vol*. 17, pp. 777-793, 2009

[10] J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York:

Plenum, 1981.

[11] R. Krishnapurum and J.M. Keller, "A possibilistic approach toclustering", *IEEE. Trans. Fuzzy Systems*, vol. 1, pp. 98-110, 1993.

[12] N.R. Pal, K. Pal, J.M. Keller, and J.C. Bezdek, "A possibilistic fuzzy c-means", *IEEE Trans. Fuzzy Systems*, vol. 13, pp. 517-530, 2005.

[13] J. Zhang and Y. Leung, “Improved possibilistic C-means clustering algorithms,” *IEEE Trans. Fuzzy Systems.*, vol. 12, pp. 209–217, 2004.

[14] I. Gath and D. Hoory, "Fuzzy clustering of elliptic ring-shaped clusters", *Pattern Recognition Letters*, vol. 16, pp. 727-741, 1995.

[15] N. R. Pal, K. Pal, and J. C. Bezdek, “A mixed c-means clustering model,” *IEEE Int’l Conf. Fuzzy Systems*, Spain, pp. 11–21, 1997.

[16] R. Krishnapurum, O. Nasraoui, and H. Frigui, "The fuzzy c spherical shells algorithm: A new approach", *IEEE Trans. Neural Networks*, vol. 3, pp. 663-671, 1992.

[17] I. Gath and D. Hoory, "Detection of elliptic shells using fuzzy clustering: application to MRI images", *Proc. Int'l Conf. Pattern Recognition*, vol. 2, pp. 251-255, 1994.

[18] X.-H. Wu and J.-J. Zhou, “Possibilistic fuzzy *c*-means clustering model using kernel methods,” *Proc. Int’l Conf. Comput. Intell. Model., Control Autom.*, pp. 465–470, 2005.

[19] A. Guill’en, I. Rojas, J. Gonz’alez, H. Pomares, L.J. Herrera, O. Valenzuela, and A. Prieto. "A Possibilistic Approach to RBFN Centers Initialization." *Lecture Notes in Computer Science*, vol. 3624, pp. 174–183, 2005.

[20] D. Chen, D.-W. Cui, and C.-X. Wang, "Weighted Fuzzy C-Means Clustering Based on Double Coding Genetic Algorithm," *Lecture Notes in Computer Science*, vol. 4113, pp. 622 – 633, 2006.