基於 PSO 演算法之改良式碎形影像壓縮

An Improved Approach in PSO-based Fractal Image Compression

郭孟迪、黃玄煒
國立台灣海洋大學電機工程研究所
e-mail: {M96530060, B0136} @mail.ntou.edu.tw

摘要

因傳統使用徹底搜尋法之碎形影像壓縮的編碼速率太慢，本文以快速粒子群聚最佳化(PSO)為理論基礎，利用高斯函數所產生的適應函數來加速編碼流程，併合碎形四元樹分割特性的使用，將分割影像中的各個小區塊值被視為粒子，以此區塊取代歸屬於此區塊的粒子群，使其在PSO中能被導引至更合適的影像區塊。本文方法的目的在於降低傳統徹底搜尋法之龐大運算量，並加快影像壓縮的速度。

關鍵字：碎形、四元樹、粒子群聚最佳化、影像壓縮

Abstract

An approach is proposed in this paper to achieve fractal image compression based on the fractal quadtree, particle swarm optimization (PSO), and Gaussian function by using fitness function. Since the encoding speed of the traditional full search method is time-consuming, the fast evolutionary PSO approach based on Gaussian function is proposed to speed up the encoder and preserve the image quality. The PSO approach uses the properties of fractal quadtree to reduce the time-consuming full search method. In PSO, every block of the divided image is regarded as particle and to be directed to the region which is consisted of a candidate of higher similarity. Finally, the particles are replaced by the region and the purpose of fast image compression can be achieved.

Keyword：Fractal, Quadtree, Particle Swarm Optimization, image compression

一、緒論

碎形應用於影像處理的領域範圍相當的廣泛，除了描述大自然的幾何學，在影像壓縮、紋理分析、輪廓邊緣偵測，甚至是抗雜訊干擾的部份也有相關的研究，目前更結合了小波轉換進一步提升碎形本身的效能[17]。粒子群聚最佳化(Particle Swarm Optimization/PSO)則是應用在模式識別上有效的工具，可應用於各種領域中的切割、分群、分類，其目的是在一群資料中找出其共通性，以取出有意義的資訊[2,5]。相較於傳統的演算法之容易陷入局部最佳解，但缺乏取得全域最佳解的能力，PSO 演算法卻可以解決上述的問題，所需的調整參數更少，許多最佳化問題也開始利用此種進化式演算法的優點來進行問題的探索與開發[1]。但、PSO 演算法是一種隨機取樣的演算法，在較大的搜尋空間中會變得相當敏感，不易穩定收斂。Potter 將搜尋空間的分割成較小向量的方法，並證實了在執行效能方面有明

顯的改善[19,20]；有鑑於此，本文將利用碎形以進行切割並縮小搜尋範圍，以提升 PSO 的效能。

因應傳統使用徹底搜尋法之碎形影像壓縮的編碼速率太慢的問題，本文使用碎形分割與 PSO 演算法的結合[6,10]，以期達到效能的提升。若僅僅使用快速 PSO，由於沒有碎形分割的前置步驟，將導致初始的資料點數量過多；又、由於 PSO 乃是計算時間較長與收斂速度較慢的進化式演算法，當初始資料點的數量過多時，將造成計算效能的降低。所以，本文先採用兩階段式的處理，先進行碎形的四元樹分割，藉此前置步驟篩選分割圖形，以降低 PSO 所需處理的資料點，再以基於高斯函數之適應函數以進行 PSO 演算法運算。此運算中，將分割影像中的各個小區塊視為粒子，並尋找此區塊所歸屬的相似粒子群，使其能被導引至更合適的影像區塊並取代之，以達到進一步減少資料點並加速小區域之編碼流程。本文方法的目的在於降低傳統徹底搜尋法之龐大運算量，並加快影像壓縮的速度。

於下列篇幅中，第二節簡單介紹碎形與 PSO 的理論依據，第三節則提出本文所設計的架構與運算方式，第四節則是實驗的結果分析，並評估各個參數對圖形品質以及計算速度的影響，第五節總結本文方法的優點與缺點，並指出未來應該改進的方向。

二、習知技術探討

2.1 碎形幾何學

碎型影像處理最早是由 J.Hutchinson 所提出[9]，其運作原理係將影像本身分割成值域區塊(range block) 和定義域區塊(domain block)，將影像中相似的部分找出來進行壓縮，並將計算出來的參數加以記錄。其例子包含：雲、森林、星雲、樹葉，羽毛、花、岩石、山、地毯、磚等等，人眼看起來覺得很複雜，但實際上包含的資訊量卻是非常的低，因為它們可以經由重複地自我拷貝或部份的自我拷貝而產生。因此，其累贅度相當高。碎形的應用有：影像切割、 影像分析、 影像合成、 以及影像壓縮等[3,8]。碎形壓縮的基本目標與概念為搜尋相似的部分並加以分割，分割後對各個區塊進行壓縮，且儘量降低失真的程度[16,21]。當碎形將圖形分割完成之後，系統所得到的是一個個大小不同的破碎區塊，這些區塊都是需要進行壓縮的部份，可採用不同之壓縮方法給予進一步壓縮其內容。

2.2 粒子群聚最佳化

PSO 演算法，源起於鳥群社會的行為觀察所歸納出的一種方法，理論上是循著一群鳥在一個區域隨機搜尋食物，鳥群會依靠著自己本身的位置、過去的經驗，以及和群體之間交換訊息的方式去搜尋食物，PSO 即是模擬這樣的行為所設計的演算法[12,13]；如圖 2-1，紅點(particle)表示食物的位置，黑點表示鳥的位置，根據鳥個體本身的經驗與群體之間的訊息交換來設計適應函數，並依靠適應函數所計算的適應值來決定搜尋的方向，以求得適應值最佳化來達成目的。PSO 是具積極合作與訊息共享特性的一種演算法。

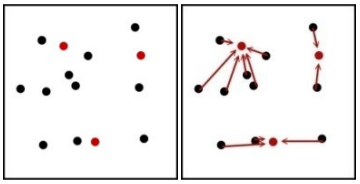
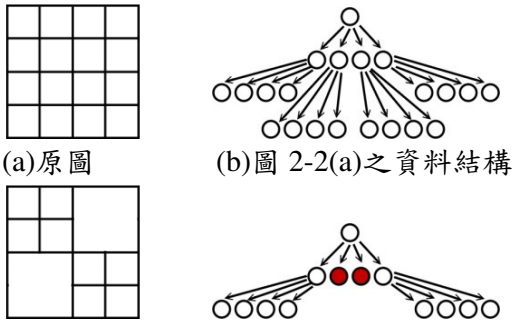


圖 2-1：PSO 示意圖

PSO 可視為一種群聚分析，群聚分析是一種將相似資料歸類的技術[19]。群聚分析的目標是讓每一個群聚之中的成員彼此間資料特性的相似性達到最高，而與其他群聚的相似性達到最低。假設每個個體均有最高適應度的自我最佳經驗($pbest_p$)；相對於個體，整個群體的全域最佳經驗稱為 gbest。以前述鳥群為例，距離某一個食物位置近的鳥群歸為同一個資料群，而計算與食物的距離將是設計經驗函數(適應函數)的重要指標。透過個體的經驗彼此交換與差值，可得到而每一

個群聚的中心點(Cluster Center)，亦即該群聚中成員資料的中心值；求群聚中心的方法有很當的多，在本文我們採用一個簡單又普遍的方法 K-means[14]。由初始群中心的設定，至計算並歸屬資料點屬於哪個群中心，到最後重新更新群中心位置後，再次進行下一次迭代計算至群中心穩定收斂為止，其演算法流程如下節之圖 3-1 所示，更詳細的運算方式亦將揭示於下一節。

以圖 2-2 為例，藉由碎形分割影像之後得到多個區塊，這些區塊就如同圖 2-2 中的黑點，而紅點就是我們想要找到的群聚中心點；當我們找到這些紅點的區塊後，便把歸屬於該紅點的黑色區塊通通以紅點區塊取代，以達成壓縮影像的目的。如圖 2-2 所示，(a)包含(b)中的 21 個資料點，經過 PSO 計算找到各自歸屬的紅點後，取其壓縮結果如圖(d)所示，資料點縮減為 13 個。PSO 的工作即是在找出可取代這些黑點位置的灰階值。



(a)原圖　(b)圖 2-2(a)之資料結構

(c)經 PSO 壓縮　(d)圖 2-2(c)之資料結構

圖 2-2：資料點之數目變化

三、改良式快速 PSO 之碎形影像壓縮

於本節中，3.1 提供我們的系統架構與處理程序，3.2 則針對碎形四元樹與快速 PSO 的整體設計分析提供說明，3.3 乃是分析參數設定的考量與方法。

3.1 系統架構

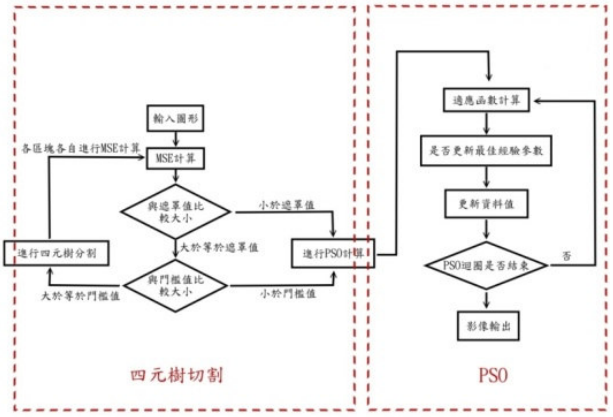


圖 3-1：系統架構流程圖

本文影像壓縮系統架構如圖 3-1 所示，其中包含四元樹切割與 PSO 壓縮程序兩部份;其中，圖形影像大小需為 $2^n x 2^n$ (n 為正整數)。

於四元樹切割部份中，先將圖形影像轉為灰階影像，再評估影像是否需進行分割；而分割的依據乃是取決於影像區塊的複雜度[7,11]，我們採用均方誤差 Mean Squared Error (MSE)[22]，以評估影像區塊的複雜度。區塊的 MSE 值大表示該區塊範圍中的像素灰階值變化較大，而區塊的 MSE 值小則表示該區塊中的像素灰階值變化較小；亦即，若區塊中是屬於相似的影像，則其灰階值變化應會較小。

當區塊進行 MSE 計算後，其平均灰階值大於 MSE 門檻值參數 threshold 時，則將該區塊進行四元樹分割；若平均灰階值小於或等於 MSE 門檻值參數 threshold，則將該區塊進行 PSO 運算。參數 pso_mask 代表了影像中做 PSO 處理之最小區塊門檻值，可由使用者決定其值；當區塊進行 MSE 計算後，若該區塊的邊長小於 pso_mask 時，則不再進行四元樹分割。由於四元樹架構乃是採用遞迴的方式進行，系統會將符合門檻值與遮罩大小的區塊資料各自進行 PSO 運算。

3.2 改良式快速 PSO 之碎形壓縮設計

PSO 碎形壓縮有三個主要功能與考量，其中

包括：PSO 適應函數、PSO 最佳經驗參數與 PSO 更新速度參數 v_p。一如 MSE，PSO 適應函數乃是用於找出相似的區塊以取代這整個區塊群組。本文採用高斯分佈來做為適應函數設計的依據，以尋找相似的影像區塊；如圖 2-2 所示，相似的區塊會由群聚中心取代，以達到壓縮的目的。

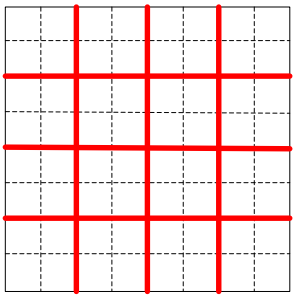


圖 3-2：pso_mask=2 的圖形分割

於圖 3-2 中，對圖形分割的 16 個區塊進行高斯分佈計算，會產生出 16 個數值 f_1~f_{16}，將這些數值相加以求得其適應值；高斯分佈計算方式如(1)所示，我們稱之為適應函數，其中 x_i 表影像的區塊灰階值，$\bar{y}$表所有影像區塊 x_i 的平均灰階值，σ 為標準差。

$$f(x;\mu,\sigma)=\frac{1}{k_0+\sigma}\sum_{i=1}^{n}\exp\left(\frac{-(x_i-\bar{y})^2}{k_0+2\sigma^2}\right) \quad (1)$$

與習知高斯函數相比較，(1)式捨棄了高斯函數的常數$(1/\sqrt{2\pi})$。其原因在該常數於迭代運算過程中對適應值影響不大，捨棄之可加快計算速度。又當區塊中每個像素的數值越接近其平均灰階值的時候，則其指數函數(exp)中的分子會趨近於 0，亦即、當圖形區塊為一顏色均勻的影像時，指數函數中的分母會趨近於 0，故我們在式子(1)中增加極小值之參數 k_0，以避免產生無意義的(0/0)運算。

3.2.2 PSO 最佳經驗參數

經過適應函數的計算過後，本文以習知式子(2)、(3)來進行適應值之大小比較並更新該值；其中、第一次進行 PSO 運算時，所計算出來個體適應值即為最佳自我經驗 $pbest_p$，而不進行式子(2)、(3)的比較。gbest 所表示的乃是群體最佳經驗，本文採用 K-means[14]將所有的個體自我最佳經驗 $pbest_p$ 的群聚中心找出來，以此作為 gbest 之值。

$$\mathbf{pbest_p^{t+1}=max(x_p^{t+1},pbest_p^t)} \quad (2)$$

$$\mathbf{gbest^{t+1}=max(pbest_p^{t+1},gbest^t)} \quad (3)$$

3.2.3 PSO 更新速度參數v_p

經更新 $pbest_p$ 與 gbest 後，接下來所進行的是習知式子(4)、(5)中之位置 x_p 與速度 v_p 數值更新。於原始 PSO 中，x_p 代表了鳥群的位置與 v_p 代表了鳥群飛行的速度；當鳥群所搜索的範圍較廣，或是滿足一些條件(如：事前知道食物離鳥群較遠之類的)，此時可以提升初始的速度 v_p，以加快鳥群搜索的速度。相同的，當以影像圖形做為考量時，x_p 代表了影像圖形的灰階值，而 v_p 則以灰階值的變化量來代表。當影像中灰階變化較大時，v_p 值則隨之上升，以加快搜尋速度；當影像中灰階值變化較小時，v_p 值則隨之下降，使得搜尋的結果更加精確。

$$\mathbf{v_p^{t+1}=\tau\cdot v_p^t+c_1*rand_a*(pbest_p^t-x_p^t)+c_2*rand_b*(gbest^t-x_p^t)} \quad (4)$$

$$\mathbf{x_p^{t+1}=x_p^t+v_p^{t+1}} \quad (5)$$

3.3 參數分析與設定

五個參數將影響我們的影像壓縮結果，包括：c_1、c_2、τ、threshold、pso_mask；其中：c_1與c_2為用來$pbest_p$與gbest影響比率的參數、τ決定迭代過程中的速度v_p變化的幅度、threshold表示四元樹分割中門檻值的大小、以及pso_mask表碎形分割中遮罩的最小值。

threshold 表示四元樹分割中門檻值的大小，此為最初碎形分割所使用的參數。當參數過大時，圖形的分割會過分粗糙，導致無法辨識其原貌；同樣地，當其值過小時，圖形則過度分割，導致壓縮效率不彰，其值約設定在 100~500 之間。

若程式中的參數 pso_mask 設為 2，表示當四元樹切割至 2×2 的矩陣時，將會不考慮 MSE 值的大小，而強制進行 PSO 計算。若將 pso_mask 的值設為 1，因只包含了一個像素的灰階值，雖然仍可計算，但其資訊量太低，會使的系統的效率下降，且圖形壓縮結果差異不大，故我們的最小值通常設為 2。

由於我們影像壓縮程序是採用先以碎形四元樹的切割方式，以降低計算的複雜度。由於 c_1 與 c_2 分別控制個體自我最佳經驗與群體自我最佳經驗，故其數值的設定，表示在處理壓縮的情況下將以整體區塊的顏色為主要考量方向；若有少數像素出現差異較大的顏色時，將以整體為主。當圖形出現如圖 4-4 的雜訊時，此時將 c_1 設為 0.2 與 c_2 設為 0.8，將可以減低雜訊對降低壓縮效率所造成的影響。

τ 用以設定迭代過程中速度 v_p 的變化幅度，當 τ 值設定過大時，將會使的圖形出現白色碎塊的情況；這是由於在 8bit 灰階的圖形中，255 所表示的是白色。如式子(4)、(5)所示，PSO 演算法中的速度 v_p 是採用一種累加的算法；即使 c_1 與 c_2 參數部份的算式影響很小，但由於每次累加之 $\tau \cdot v_p^t$ 數值過大，將導致其灰階值過度增加，而使圖形出現白色區塊。故本文將 τ 值設為 0.1，即表示一張影像中以區塊為單位，計算出其標準差，以其標準差大小的十分之一為變化幅度，將使碎塊出現的機率降低。藉由本文所設計之適應函數做進一步評估，當適應函數變化量過小時，則終止 PSO 的迭代計算，如此不但可以加快計算速度，更可以避免 τ 值設定過大所產生之白色碎塊。

四、實驗分析

本章節將對本文系統程序進行實驗與分析，所採用的測試圖形為一般影像處理常用的測試圖形，其來源為其來源為美國南加州大學訊號與影像處理研究所[15]；限於篇幅，我們只列出部份結果，更詳細的內容可見於[4]。4.1 節針對一般影像處理常用的圖形照片，進行測試與分析，4.2 節則以五個重要參數進行分析，測試其對影像圖形的影響與壓縮速度的變化。4.3 節則對紋理圖形進行實驗。本文以 PSNR (Peak Signal to Noise Ratio)做為評估圖形失真程度的依據，其單位為 dB。當 PSNR 值越大，就代表失真越少。一般而言，PSNR 為 30 左右，兩張圖就很接近；PSNR 為 40 左右，肉眼幾乎看不出差異；當 PSNR 為 70 左右，可以說是幾乎一模一樣。

4.1 一般影像實驗成果分析

對於一般常見的影像圖形，本文將以下列圖 4-1 中之圖形進行測試，參數的設定為 c_1=0.2、c_2=0.8、τ=0.1、threshold=100、pso_mask=4。

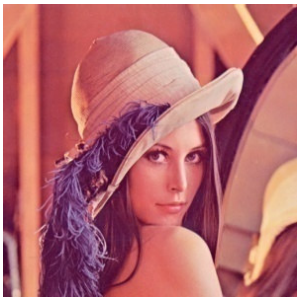


圖 4-1：512×512 測試圖形

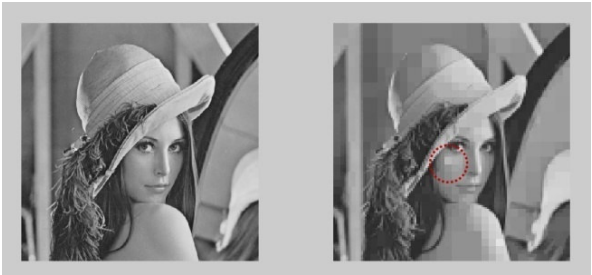
圖 4-1 中的影像以 tiff 無失真影像檔為圖片來源，經轉為灰階後，實驗結果為壓縮比率= 2.6%, PSNR=26.4303, 2.98seconds，如圖 4-2 所示。亦即、影像讀取時其檔案大小為 512×512 uint8，經過壓縮後，其檔案大小為 6805 unit8，故壓縮比率為(6805/(512*512))=2.6%；表示經過壓縮後，其資料結構的節點數只佔原圖節點數 2.6%。上述的壓縮比率亦表示，當檔案大小為 256KB，經過壓縮後其檔案大小約為 6.64KB。圖 4-2 中，有許多大小不同的區塊，以紅色虛線表示之，此大小不同之區塊即為碎形四元樹切割的成果，而區塊中的灰階顏色則取決於 PSO 演算法的計算。

圖 4-2：壓縮成果

4.2 參數對實驗成果分析

4.2.1 參數 c_1 與 c_2 之分析

由於本文採取碎形切割的前置步驟，所以區塊中顏色差異範圍被限制在門檻值內。若每個像素的顏色較相近，則參數 c_1 與 c_2 對於圖形壓縮影響的較小；但若像素顏色差距較大，則 c_1 與 c_2 的不同設定，將造成壓縮結果產生重大差異。當 c_1 遠大於 c_2 時，表示以個體為主要考量方向，而 c_2 遠大於 c_1 時，則表示以群體為主要考量方向。



(a)以個體為主要考量較易產生灰階值的變化

(b)以群體為考量灰階值變化較不易產生

圖 4-3：c_1 與 c_2 測試圖形

如圖 4-3(a)中，以 c_1=1.99、c_2=0.01、τ=0.1、threshold=100、pso_mask=4 之設定值進行壓縮，紅色虛線部分顯示白色區塊被產生；而圖 4-3(b)以 c_1=0.01、c_2=1.99 之設定，則產生較均勻之影像色差。事實上，執行壓縮後的影像大小會相等，但影像之像素灰階值未必會相同，但若以 c_2 遠大於 c_1 為設定參數的主要考量時，壓縮產生相同結果的機率較大。當將圖形加入雜訊時、亦具有相同狀況；亦即，c_2 遠大於 c_1 時，雜訊對於圖形壓縮所產生的影響較小。如圖 4-4 所示。

(a)　c_1=1.99、c_2=0.01

(b)　c_1=0.01、c_2=1.99

圖 4-4：雜訊壓縮測試圖形

4.2.2 參數 threshold 與 τ 之實驗分析

基於本文程序之壓縮，threshold 與壓縮比率成正比，threshold 越大則其壓縮效能將會提升，但影像解析度也將隨之下降。其他參數設定為 c_1=0.2、c_2=0.8、τ=0.1、pso_mask=4，實驗結果如圖 4-5 所示：(a) 5%壓縮比率、PSNR=26.9085、4.87seconds；(b) 3.24%壓縮比率、PSNR=26.6873、3.48seconds；(c) 2.6%壓縮比率、PSNR=26.4303、2.98seconds；(d) 2.05%壓縮比率、 PSNR=25.8195、2.66seconds。

(a) threshold=10

(b)threshold=50

(c) threshold=100

(d)threshold=200

圖 4-5：threshold 對影像壓縮之影響

以下將 threshold 之值設為 10~200 進行實驗，實驗結果下圖 4-6 所示，壓縮時間與 threshold 成反比。亦即、當 threshold 值越大，則其壓縮所花費的時間越少；當 threshold 值越小，則其壓縮所花費的時間則越多。而圖 4-6 中，threshold 為 90、140、170 時，運算時間有稍微上升的情況，此乃是由於 PSO 演算法中包含了隨機變數所產生之影響。

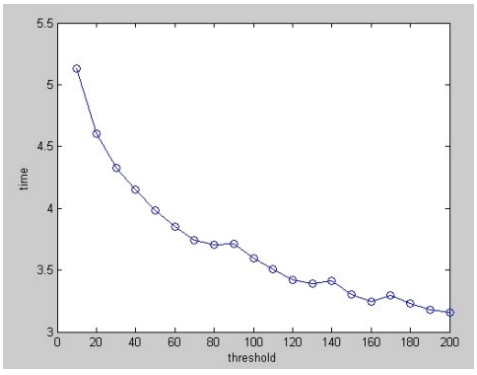


圖 4-6：threshold 對壓縮時間之影響

接著進行參數 τ 之實驗分析，首先將其他的參數固定為 c_1=0.2、c_2=0.8、threshold=100、pso_mask=4，再變更參數 τ 進行分析如下圖 4-7 所示：

(a) τ=0.05/8.29seconds PSNR=26.4256

(b) τ=0.1/8.31seconds PSNR=26.3973

(c) τ=0.3/8.33seconds PSNR=25.7488

(d) τ=0.6/8.53seconds PSNR=18.2247

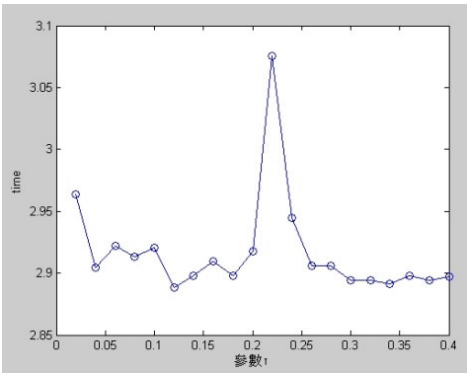
4-7：參數 τ 對影像壓縮之影響

由圖 4-7 中可瞭解，由於式子(4)乃是屬於累加計算，過大的 τ 值將會產生如上圖 4-7(c)/(d)中的白色碎塊。為了改善此情況，本文以適應函數的變異量做為離開 PSO 迴圈的評估條件後，不但改善了白色碎塊的問題，更大幅提升了計算速度，如圖 4-8 所示：

圖 4-8：τ=0.6、2.92seconds、PSNR=26.4305

由於參數 τ 控制著 v_p 變化的幅度，而 v_p 值除了受到參數 τ 之影響外，也受到最佳經驗解的控制，不斷進行值的更新。最佳個體經驗 $pbest_p$ 和全體最佳經驗 gbest 差異較大時，v_p 值將上升以增加收斂速度；反之、則降低收斂速度以避免過度計算，故其壓縮時間變化不大，其計算時間約在

3.1~2.85 之間，如下圖 4-9 所示。而當參數 τ 為 0.22 所計算的時間較長的原因，也是由於受 PSO 演算法中隨機變數的影響。



4-9：參數 τ 對壓縮速度之影響

4.2.3 參數 pso_mask 之分析

將參數設為 c_1=0.2、c_2=0.8、threshold=100、τ=0.1，以 pso_mask 為 1、2、4、8 進行實驗，如下圖 4-10 所示，(a) 13%壓縮比率、PSNR=32.3487、12.96seconds；(b) 13%壓縮比率、PSNR=29.6705、6.37seconds；(c) 2.6%壓縮比率、PSNR=26.4305、2.92seconds；(d) 0.95%壓縮比率、 PSNR=23.5187、1.51seconds。

(a) pso_mask=1　　(b) pso_mask=2

(c) pso_mask=4　　(d) pso_mask=8

圖 4-10：pso_mask 之參數分析

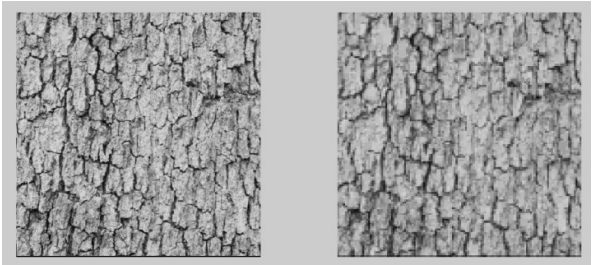
由上面實驗可知，當 pso_mask 為 1、2、4 時，其影像壓縮結果差異不大，但其計算時間卻有巨大差異；而當 pso_mask 為 8 時，雖然其計算時間約為 pso_mask 為 2 時的一半，但是影像已經出現模糊的情況。下列表一列出各個參數與影像壓縮的關係。

表一：各參數與影像壓縮的關係

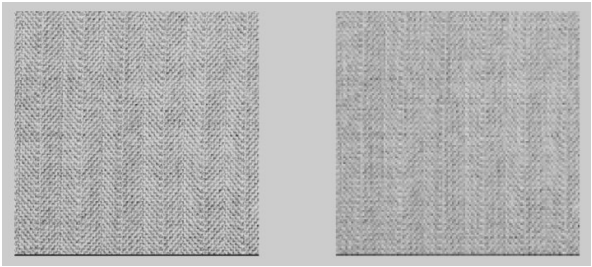
參數 c_1 與 c_2	控制影像編碼以個體或整體最佳經驗為依據。
參數 τ	影響著灰階值變化的幅度，過大的設定將造成白色碎塊的情況。
參數 threshold	此數值為碎形分割條件之門檻值，其值上升，會使得壓縮效果上升、失真程度上升，但計算時間下降；反之數值下降，壓縮效果下降、失真程度下降，但計算時間上升。
參數 pso_mask	此參數上升，會使得壓縮效果上升、失真程度上升，但計算時間下降；反之數值下降，壓縮效果下降、失真程度下降，但計算時間上升。與 threshold 不同的是，此參數控制著最小區塊的大小。

4.3 紋理圖形成果分析

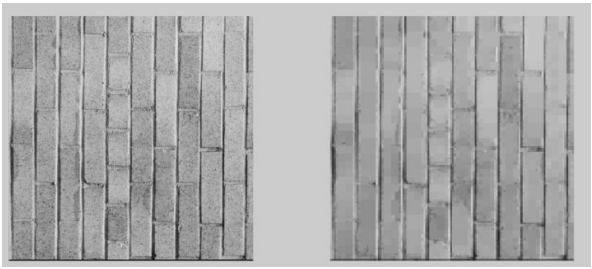
最後本文將對設計架構程序時，未考慮到的紋理圖形進行實驗測試，由於紋理的圖形的特性是紋路的部份與區塊的部份差異值較大，故提高 threshold 值以增進壓縮效率，其參數設定為 c_1=0.2、c_2=0.8、threshold=300、τ=0.1、pso_mask=4，測試結果如圖 4-11 所示。



(a) 5.61%壓縮比、5.32seconds、PSNR=19.6055



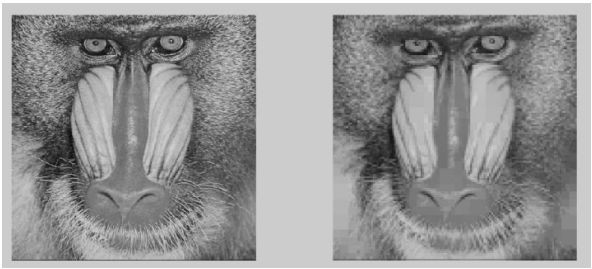
(b) 6.19%壓縮比、5.76seconds、PSNR=20.5421



(c) 2.62%壓縮比、2.92seconds、PSNR=22.3690

圖 4-11 紋理壓縮測試

由上圖 4-13 (a)/(b)可瞭解，相對於一般的影像，紋理影像所壓縮的時間較長，壓縮的比率較差，而圖 4-13(c)乃是磚牆的紋理，此種紋理形狀符合四元樹切割的結構形狀，故其壓縮效能較佳。最後測試紋理與一般圖形混合的圖形，如下圖 4-12 所示，threshold=100、其他參數同上：



4-12：含有紋理的圖形測試分析

花了 4.89seconds 得到 5.04%的壓縮比率，以及 PSNR=20.7263。當處理含有紋理的影像圖形時，對於其沒有紋理的部分可以進行較佳的碎形壓縮，而對於其有紋理的部份則須以 pso_mask 為單位進行壓縮，無法有效的以碎形進行壓縮；或、採取不同之碎形切割方式[18]。

五、結論

對於影像壓縮的研究，本文在此採用了兩個理論基礎以建立影像壓縮的架構，其中碎形乃是常被應用於影像壓縮的理論之一，而 PSO 為一個模式識別方法，其可應用於碎形中影像編碼壓縮的部份。經過本文的實驗分析後，可以瞭解到對於一般影像的壓縮效能較佳；但是對於具有紋理的影像壓縮效率較不佳，此乃是由於碎形在進行四元樹分割時，僅以圖形變化的複雜度做為分割的依據，故未來若加入紋理辨識的功能，對於壓縮的效能將可提升。另外，碎形切割的方式也可更進一步採用不規則分割的形式，以便將不規則形狀之紋理能進行更合適的切割方式，藉以提升壓縮效能。

六、參考文獻

[1] 王良吉，“應用 PSO 演算法於分類法則之探勘”，國立高雄第一科技大學碩士論文，Jan. 2007。

[2] 張仕政、陳大正，“利用合作型粒子群演算法於非線性工程設計最佳化問題之應用”，台灣作業研究學會 2007 年學術研討會暨年會，國立東華大學，Dec. 2007。

[3] 吳文成，“碎形 Fractal”，http://www.atlas-zone.com/complex/fractals/index.html#new

[4] 郭孟迪、“以改良式快速粒子群聚最佳化之碎形影像壓縮”、國立台灣海洋大學碩士論文、June 2009。

[5] 黃建彰，“Genetic Algorithm with Particle

Swarm Optimization for Multiple Sequence Alignment”，義守大學碩士論文、2008。

[6] 曾俊傑，“基於粒子群聚最佳化演算法之碎形影像壓縮及主動輪廓模型”，國立中山大學博士論文，Aug. 2008。

[7] 曾修宜、黃文宣、楊智宇、劉濬毅，“以四元樹為基礎抽取影像特徵之影像檢索技術”，TANET 2006 台灣網際網路研討會，花蓮，2006。

[8] 廖思善，“碎形密碼”，科學月刊，2001。

[9] 蔡義道，“區塊間相關性搜尋策略之碎形影像及視訊編碼之研究”，國立成功大學碩士論文，1999。

[10] 謝昇達，“Improvements and Applications of Population-based Optimizers”，國立台灣師範大學博士論文，July 2007。

[11] 顧叔財、胡育誠，“一個植基於向量量化編碼法之影像認證技術”，2003 民生電子研討會，台南，Nov. 2003。

[12] R. C. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” Proc. Sixth International Symposium on Micro Machine and Human Science, pp.39-43, 1995.

[13] R.C. Eberhart and Y. Shi, “Particle Swarm Optimization: Developments, Applications and Resources,” Proc. IEEE Int. Conf. On Evolutionary Computation, Vol.1, pp. 81-86, 2001.

[14] K-means clustering algorithm,” Applied Statistics, 28, pp. 100-108, 1979.

[15] http://sipi.usc.edu/database/index.html

[16] A. E. Jacquin, “Image coding based on a fractal theory of iterated contractive image transformations,” IEEE Trans Image Process 1992, 1, 18–30.

[17] A. Malviya, “Fractal based spatial domain techniques for image de-noising Audio,” ICALIP 2008, pp. 1511–1516, 2008.

[18] S. Mozaffari, K. Faez, M. Ziaratban, “Character Representation and Recognition Using Quadtree-based Fractal Encoding Scheme,” Proceedings. Eighth International Conference on Document Analysis and Recognition, pp. 819-823, Sep. 2005.

[19] M. A. Potter and A., K. Potter, “ A Cooperative Coevolutionary Approach to Function Optimization,” Lecture Notes in Computer Science, 866, 1994.

[20] F. Van Den Bergh and A. P. Engelbrecht, “A Cooperative approach to particle swarm optimization,” Evolutionary Computation, IEEE Transactions on, 8(3), pp. 225-239, 2004.

[21] C. C. Wang, L. C. Lin and S. H. Tsai, “Fast fractal encoding algorithm using law of cosines,” IEEE 47th MWSCAS, pp.233-236, Hiroshima Japan, 2004/07.

[22] Wikipedia, “Mean squared error,” http://en.wikipedia.org/wiki/Mean_squared_error