應用動態式自我組織神經網路於骨架擷取之研究

Application of Dynamic Self-Organizing Map in Skeleton Extraction

張鴻德
南台科技大學資訊工程系
hdchang@mail.stut.edu.tw

廖俊傑
南台科技大學資訊工程系
m93g0206@webmail.stut.edu.tw

摘要

骨架擷取技術在許多應用領域中被廣泛地採用。而過去傳統的細線化(thinning)所產生的骨架，往往會多出不必要的小分支或在叉點處產生變形，造成在特徵擷取上的不穩定。因此動態式自我組織神經網路(Dynamic self-organizing map,DSOM)被提出以擷取出更為完整的骨架。但實際應用上卻發現執行的速度較為緩慢，且其在處理具有叉點(fork)及迴圈點(loop)的圖形時費時更久。因此，本研究提出利用端點估測及影像分割的方法來加快處理速度。端點估測可以使得輸出神經元一開始便置於圖形的骨架點上，而不會落在圖形的空白處；影像分割則讓神經元在增減時不需檢查是否為叉點及迴圈點以節省時間。在實驗部份則利用工具、數字、英文字母及中文字等圖樣作為實驗對象。從實驗結果可以發現，比較僅用 DSOM 以及加上本研究所提的方法，其骨架擷取結果相同，但後者速度可以提升19%。

關鍵詞：骨架擷取、文字辨識、細線化、自我組織神經網路。

Abstract

Skeletal shape extraction technique was widely adopted in many applications such as object modeling、character recognition and computer animation. The thinning process always be used in the skeleton extraction, but it often distorts the local shapes of an observed pattern. This is an inherent defect for all thinning algorithms. Therefore, a Dynamic Self-Organizing Map (DSOM) was proposed to extract skeleton precisely. But the process speed of DSOM is too slow. In order to speed up the process speed, the methods of endpoints estimation and segmentation were proposed in this paper. Endpoints estimation process estimates the position of output neurons instead of random decision. The image segmentation process avoids the fault when DSOM used in crossing or loop pattern. In the experimental results, the tools、alphanumeric and Chinese characters are used as test patterns. The results show that the proposed methods improve 19% of DSOM process speed.

Keywords：skeleton extraction、alphanumeric recognition、thinning、self-organizing map.

一、前言

影像的形狀分析不管在圖訊(pattern)的辨認、形狀的形成、以及電腦視覺等應用裡，都是最基本所要面臨的問題。在過去的研究中，也有藉由抽出其骨架而來代表此形狀的分析方法。而如此的方法能夠將所必要的資料大量的精緻化，僅藉由骨架就能夠表達出此形狀。因此在許多的應用領域裡也被廣泛地採用，如物件模組化、影像辨識、光學字元辨識、醫療影像的分析以及印刷電路版的檢驗[1]等應用領域。而傳統的骨架擷取技術[2]往往會在端點處產生許多不必要的小分支，以及在叉點處產生變形。而這些情況容易造成特徵擷取

上的不穩定，進而影響到影像辨識及分類時的錯誤。因此，如何擷取出一個正確的骨架，已經是極為重要的課題。

目前一般所使用的骨架化方法為由T.Y.Zhang及C.Y.Suen所提出的快速式平行細線化演算法[3]。此方法最重要的特色是其速度非常快且程式碼易於實作。然而，此種方法對於雜訊的影響非常敏感。往往由於雜訊的干擾，使得細化後的圖形產生一些不必要的分支。且其和其它細化演算法一樣，在叉點處會產生變形。而當輸入的影像過於稀疏時，此方法也無法正確擷取出骨架。雖然後續對於此方法的改良非常多[4][5][6]，卻依然無法完全改善其缺點。

近年來，由於類神經網路技術上的成長，使得其被應用在各種領域裡。其中，由T.Kohonen所提出的自我組織映射圖網路(Self-organizing map,SOM)[7]被應用於骨架擷取[8]。其基本流程是學習前先行預估所需的輸出神經元數目，再將其輸入到SOM網路裡進行學習。在學習的過程中，這些輸出神經元會不斷的與輸入圖形的像素點做相似性量測(即距離比對)，再由SOM網路的學習演算法做更新，直到整個網路達到收斂的條件為止，骨架即形成。但要如何準確的預估所需輸出神經元的數目是極為不易的。過多的神經元會造成網路的過度學習，使得網路的學習速度變得極為緩慢；而不足的神經元則會使得所形成的骨架無法完整表達輸入圖形所代表的意義。

基於上述原因，A.Datta及S.K.Parui提出了一可動態調整輸出神經元的動態式自我組織神經網路[9][10]。此網路可在學習過程中，增加或刪減輸出神經元，以使得輸出神經元數目能達到輸入圖形所需的數量。然而在實際應用上卻發現此網路執行的速度不夠理想。因此，本研究針對此問題提出利用端點估測以及影像分割的方法來做改良。在實驗的部份，本研究以工具、數字、英文字母及中文字等圖樣作為實驗對象。從實驗結果可以發現，本研究提出的方法可以有效的提升動態式自我組織神經網路的速度。

二、系統架構

動態式自我組織神經網路是一可在學習過程中任意增減輸出神經元的網路。首先在前處理的部份利用端點估測的方法來偵測輸入圖形的端點，並將這些端點當作初步的輸出神經元，接著將其輸入DSOM網路。首先利用傳統SOM網路的演算法進行學習，待其學習完成之後，即利用DSOM演算法進行神經元的增減，直到網路收斂為止。但此時仍是無分支的骨架，所以需將神經元連結起來。然而，若不事先規劃神經元連結的方向，則容易使得骨架產生變形。因此，本研究利用影像分割的方法，將影像劃分成若干區域，讓彼此區域相鄰的神經元連結，以防止骨架變形。

(一) 前處理

1. 隨機估測

在使用SOM網路進行骨架擷取之前，我們必需先行估算學習過程中所需輸出神經元的數目。然而，這是一個非常困難的步驟。這主要是因為每一張輸入圖形的大小、形狀皆不同，因此所需的輸出神經元數目自然也就不一樣。但是到目前為止，尚未有一個方法能完整且正確的估測出骨架點(輸出神經元)所應在的位置與數目。這使得應用SOM網路於骨架擷取上，產生了重大的困難。但由於動態式神經網路的提出，於是目前對於一般輸出神經元便採用隨機估測的方式，也就是不管圖形的大小及形狀，任由系統隨機估計神經元數目及其坐落的位置。也由於這個原因，使得SOM網路的鄰域範圍(neighborhood)必須設定得非常大。由[7]中可以知道，SOM網路不同於其它神經網路最大的特點，便是它具有鄰域的特性。此特性會使得位於勝者神經元(圖1中有c符號的圓圈)鄰域中的輸出神經元往目前輸入的像素點移動。勝者神經元能移動的距離最大，而離其愈遠的輸出神經元能移動的距離就愈小。而鄰域範圍會隨著學習的次數縮減，直到只剩下勝者神經元為止，其縮減方法如圖1所示。這主要是為了使SOM網路能符合人類大腦的特性。但也正因為這點使得SOM網路的速度較為緩慢。這是因為鄰域範圍愈大，神經元所需進行的權重更新(weight adaptation)次數便愈多。因此如欲加快SOM網路的速度，便要將鄰域範圍縮小。然而，這在實際情況上有其困難。以圖2為例，圖中的黑點即為輸入圖形，紅點則為利用隨機預估所得到的輸出神經元。從圖中可以發現，所預估的輸出神經元有很多距離像素點(黑點)非常遠。當使用者所設定的鄰域範圍太小時，這些點便得不到學習的機會而無法被拉進像素點以形成骨架。傳統上，SOM網路為了解決這個問題，便必順將鄰域範圍設定得非常大。在[7]中便指出，最好的情況是設定整張輸入圖片皆為鄰域範圍。然而這會使得整個網路速度變得更慢。因此，在一般使用上，鄰域範圍的大小便設為整張圖形的一半以上。為了解決上述的情況，使得SOM網路的執行速度能加快。因此，本研究提出了以端點來當作輸出神經元的方法。

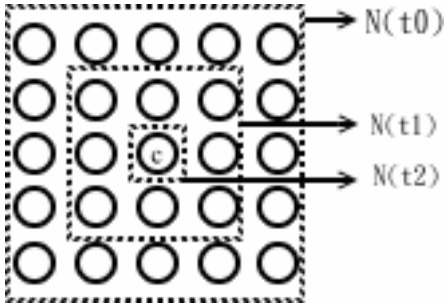


圖 1 鄰域的縮減方式

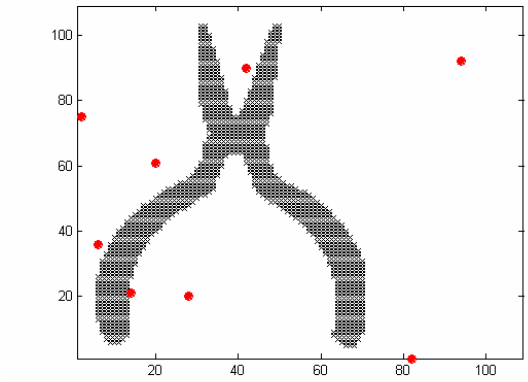


圖2 利用隨機估測所得的輸出神經元

2. 端點估測

以圖 3 為例，若想以最少的輸出神經元來形成圖形的骨架，莫過於以此圖形的端點來代表骨架點。而且利用此種方式可以保證，所得到的輸出神經元不會落於輸入圖形的像素點之外。因此，便可以縮小 SOM 網路的鄰域範圍以增加網路的執行速度。本研究利用一簡單的方法來估測端點，其流程如下：

1 讀取圖檔並二值化。

2. 由左至右，由上至下掃瞄圖形的像素點(黑點)。將每列所找到的第一點標記為藍點，最後一點標記為紅點。

3. 由左至右，由上至下計算步驟 2 中每列的藍點與紅點之距離 d。if $d < \varepsilon$，則此兩點之中點即為端點。

4. 由左至右，由上至下計算步驟 2 中每行的藍點與紅點之距離 d(歐式距離)。if $d < \varepsilon$，則此兩點之中點即為端點。

5. 結果。

以上各步驟之結果如圖 4 所示，圖 5 為針對常用的工具所做的實驗。而流程中的 ε 為一事先決定的微小正值，此值與輸入圖形的寬度有關，依據輸入圖形的寬度做動態的調整。由圖 5 中可以發現，本研究所利用的方法可以估測出大部份圖形的端點。但對於一些寬度過寬的端點無法準確估測，且會有一些不是端點的點被誤判。雖然這並不會對整個系統造成太大的影響，但這仍是本研究未來所要解決的問題。

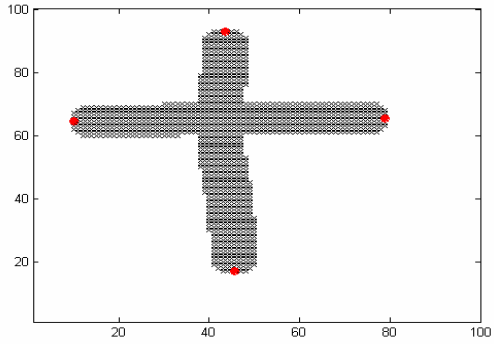
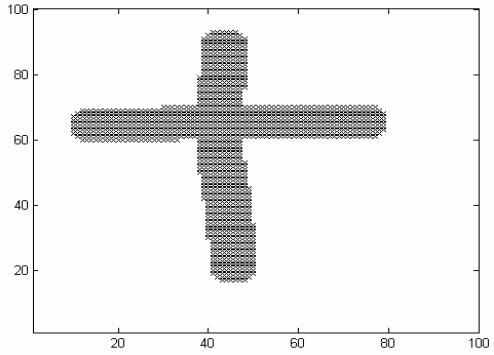
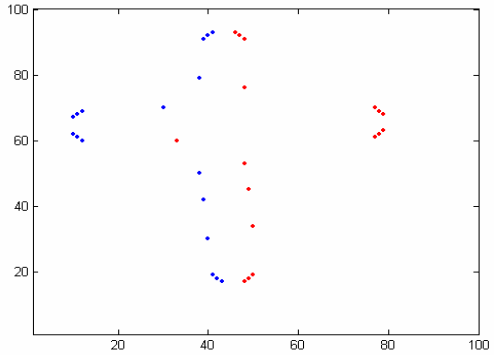


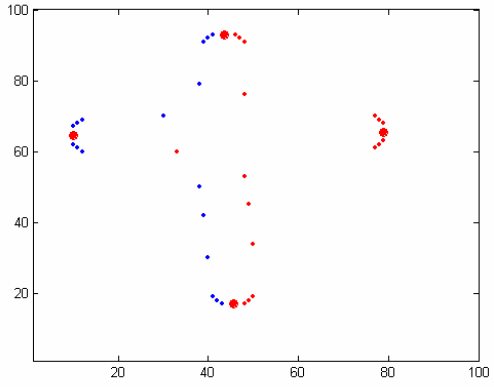
圖3 骨架形成所需的端點



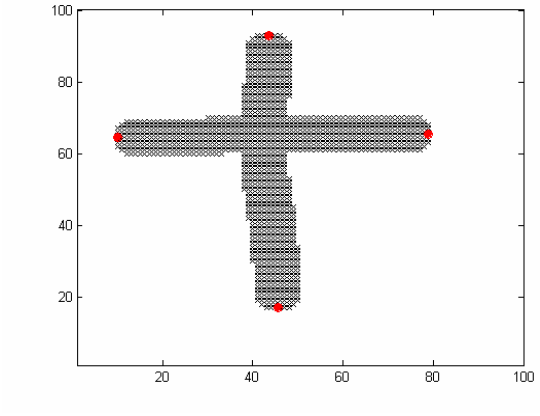
（a）輸入圖形



（b）由步驟 2 所標記的點



（c）由步驟 3、4 所得到的端點



（d）結果

圖 4 端點估測的過程

(二) SOM

當前處理完成之後，即進入SOM網路。本研究所使用之SOM網路演算法，完全依據[7]。其基本流程如下：

1. 將輸入圖形的像素當成輸入向量，表示為 $x_i = [x_1, x_2, \mathrm{K}, x_n]$，n 為像素個數。

2. 將前處理所偵測到之端點當成輸出神經元並初始化其與輸入向量之鏈結值(即距離)，表示為 $w_j = [w_{j1}, w_{j2}, \mathrm{K}, w_{jn}]$，j 為輸出神經元個數。3. 設定鄰域集合的初始大小 $N_c(t)$。

4. 針對時間 t，依序輸入輸入向量 x_i，其中 t 為一非負的整數。

5. 計算輸入向量與輸出神經元之距離 d_k。

$$d_k = \sum_{i=1}^{n} \sqrt{(x_i(t) - w_j(t))^2} \qquad (1)$$

6. 求取步驟(5)之最小 d_k，並令其為勝者神經元。

7. 更新勝者神經元即其鄰域的鍵結值。

$$w_j(t+1) = w_j(t) + \alpha(t)(x_i(t) - w_j(t)) \qquad (2)$$

其中 $\alpha(t)$ 為學習係數，初始設為 0.01，之後隨著遞迴次數 s 遞減，公式如下

$$\alpha(t) = 0.01/(1 + s/1000) \qquad (3)$$

$N_c(t)$ 隨著 t 而縮減，其縮減方式如圖 1 所示。

8. 回到步驟 4，直到 $\alpha(t)$ 或 $N_c(t)$ 其中之一收斂為止。圖 6 為尖嘴鉗經過 SOM 層學習完成之後的骨架樣式。

由圖 6 可以看出，經由 SOM 所學習完後的骨架仍不完善。因此本研究將利用俱有增減神經元特性的 DSOM 演算法加以修整。然而動態式神經網路是將符合條件的舊神經元刪除，並將新的神經元加入的動作，其動作過程中是沒有固定方向性的。因此，如不事先規劃增減神經元的方向，在遇到如叉點、迴圈點等具有兩個以上方向的特殊圖形時(如 T、X、P 等)，就有可能因為無法達到收斂條件，而使得整個系統進入無窮迴圈。因此，在[9][10]中便提出了一個偵測叉點及迴圈點的方法。然而，此種方法在實際應用上卻相當不便且速度不夠快。因此，本研究利用影像分割的方法來，來為輸出神經元規劃方向，以利於 DSOM 增減神經元。

(三) 影像分割

本研究所利用的影像分割是相當簡單的動作。其方法是依據影像中的每個像素最靠近那個神經元，則就把該像素歸類為該神經元的區域。而之所以如此做的主要原因是因為，神經元的增減處理是以兩相鄰的神經元間為主。利用影像分割可以讓我們知道此二神經元是否相連。且為了方便將影像各處的神經元連結起來形成完整的骨架。因此，本研究即利用影像分割的方法來達成此目的，圖7為一簡單圖形的例子。

(四) DSOM

在DSOM演算法的流程中，網路的更新規則與SOM層很相似。但是在學習的過程中，已存在的輸出神經元可能被刪除，而新的輸出神經元可能被新增至網路拓樸中。當兩個輸出神經元間距離過遠時，則在它們之間增加一新的神經元;反之當距離過近時，則刪除其中一個神經元。而增加或刪除輸出神經元的條件，則以(4)、(6)式為基準，(7)式為DSOM層收斂條件。

增加條件：

$$|W_k(ts) - W_{k+1}(ts)| > \sigma_1 \qquad (4)$$

新神經元的鏈結值為：

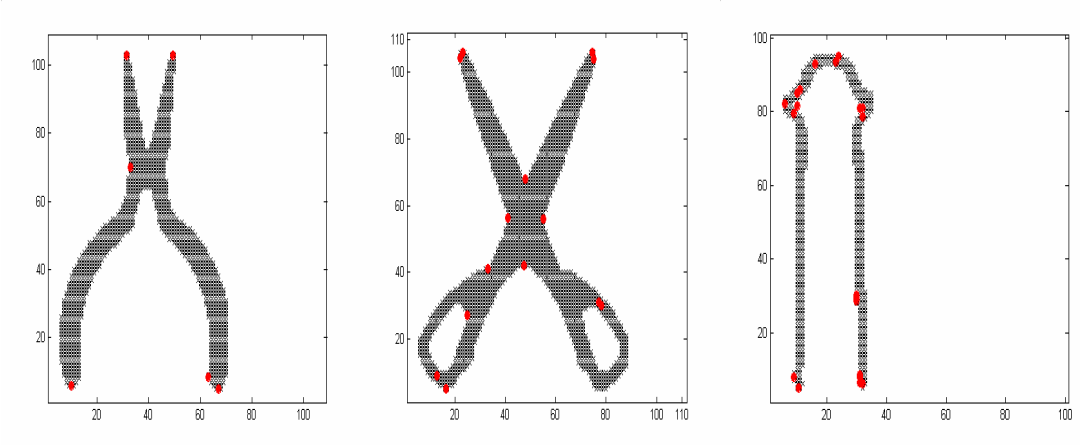
$$(W_k(ts) + W_{k+1}(ts))/2 \qquad (5)$$

刪除條件：

$$|W_k(ts) - W_{k+1}(ts)| < \sigma_2 \qquad (6)$$

收斂條件：

$$\sigma_1 > |W_k(ts) - W_{k+1}(ts)| > \sigma_2 \quad \forall\ k \qquad (7)$$



（a） 尖嘴鉗　　（b） 剪刀　　（c） 夾子

圖 5 針對各種常用工具所做的實驗

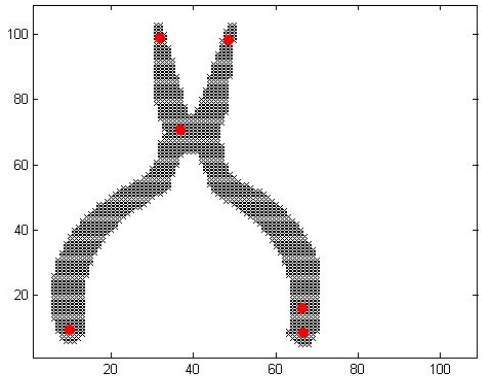


圖 6 SOM 層學習完成後的骨架

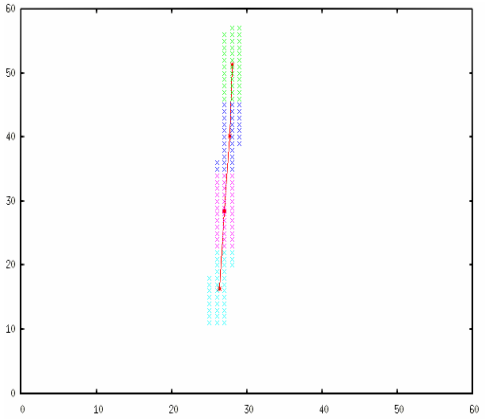


圖 7 影像分割完後的結果

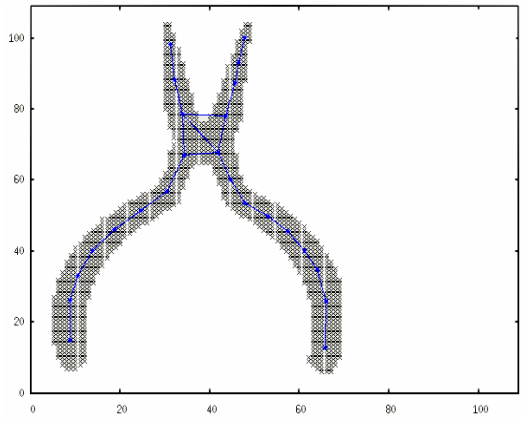


圖 8 經過 DSOM 學習完後之結果

其中 W_k 為經 SOM 層學習完後之輸出神經元，ts 為學習遞迴數。σ_1 與 σ_2 為一事先決定之正值，且 $\sigma_1 > \sigma_2$。其設定方法請參考[9][10]，DSOM 層之步驟如下所示：

1. 由左至右，由上至下決定輸出神經元之順序，以最左上之輸出神經元為始，最右下之輸出神經元為終。如果兩神經元所屬之子影像區域相鄰者，則進行處理;反之則不處理。
2. 依 (4) 式決定是否需要增加神經元。
3. 所有神經元處理完畢時，進行鏈結值更新。
4. 依 (7) 式檢查網路是否已達收斂，如收斂則網路結束。
5. 依 (6) 式決定是否需要刪除神經元。
6. 依 (7) 式檢查網路是否已達收斂，如收斂則網路結束。
7. 如果尚未達到收斂條件，則回到步驟 1。
8. 依鄰近區域將所有神經元相互連結
9. 結果。

圖 8 為圖 6 經過 DSOM 層學習完後之結果。

由圖 8 可以看出，經由 DSOM 所學習完後的骨架已相當完善。這証明了，DSOM 可以有效的改善 SOM 的缺陷。

(五) 後處理

雖然DSOM可以有效的改善輸出神經元不足或過多的狀況。然而，在圖8的中間部份可以發現，神經元之間的連結產生了錯誤。這主要是因為該點神經元的區域，同時與3個以

上的其它區域相連。因此，網路在處理的過程中便將這些點連結在一起，而產生了叉點的變形。為了解決此現象，本研究提出利用一新產生的神經元來取代變形的神經元。其處理流程如下：

1. 紀錄各神經元與其相連結的神經元有多少個。
2. 找出 3 個神經元的連結點大於 2 且相互連結者。
3. 將此 3 點合併為一新的神經元，其座標值為此 3 點的平均值。
4. 判斷是否已無滿足連結點數大於 2 且相互連結的 3 個神經元。
5. 如果還有的話再回到步驟 2 直到滿足步驟 4 的條件。

圖 9 為圖 8 經過後處理之後的結果。

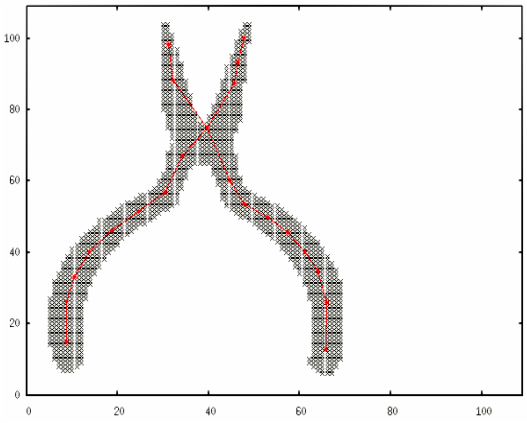


圖 9 經後處理完後之結果

由圖 9 可以觀察出，經後處理之後的骨架已經是相當完善，叉點變形的問題也已獲得解決。

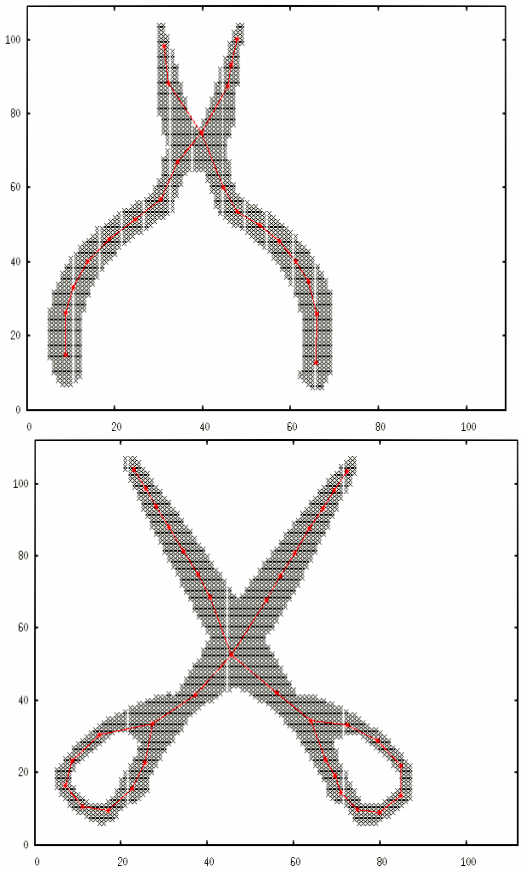
三、實驗結果及分析

以下將針對工具、數字、英文字母以及中文字的樣本作實驗，結果如圖 10 所示。表 1 則為利用隨機估測神經元與利用端點偵測及影像分割兩種方式之間速度的測試。其中工具使用最常見的尖嘴鉗、剪刀；數字則為較具代表性的 0、5、8；英文字母則從中選出擁有叉點及迴圈點的字母 A、F、X；中文字則挑選「我」與「流」兩個字。

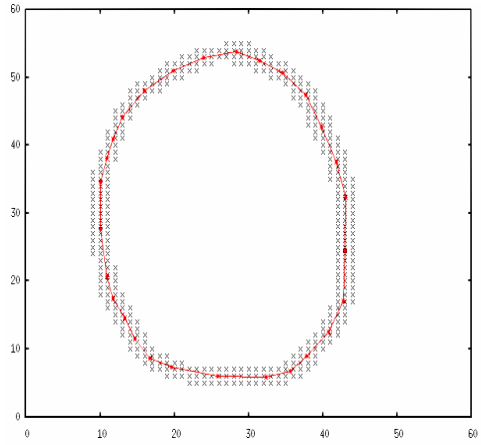
本研究所使用之軟體為 Matlab 6.5.1，硬體為一個人電腦。CPU 為 AMD Barton 2500+，記憶體為 1GB，作業系統為 Windows XP sp1，測試影像是直接用數位相機取得，並將其轉換成二值影像後輸入本系統中。實驗結果如圖 10 所示。

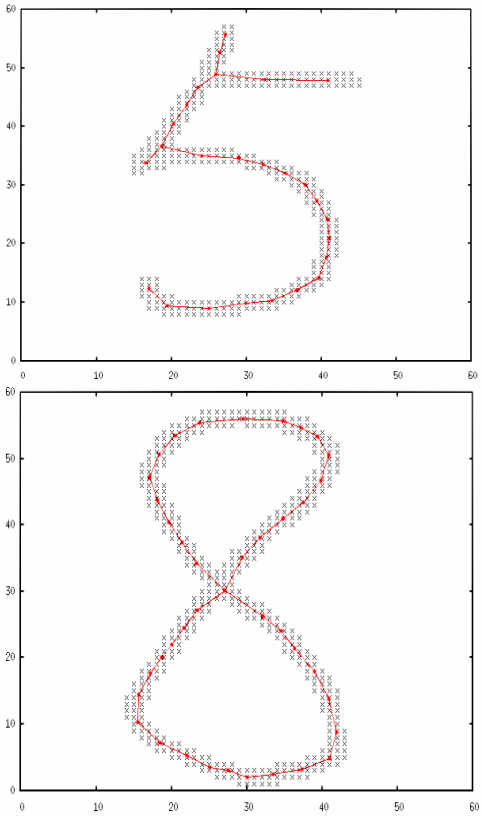
表 1 測試方法及所需使間之比較

	測試方法及所需時間	
測試圖案	隨機估測+角度偵測	端點偵測+影像分割
鉗子	54秒	43秒
剪刀	57秒	44秒
0	31秒	25秒
5	36秒	28秒
8	40秒	31秒
A	43秒	31秒
F	49秒	42秒
X	48秒	35秒
流	68秒	59秒
我	73秒	64秒
平均時間	49.9秒	40.2秒

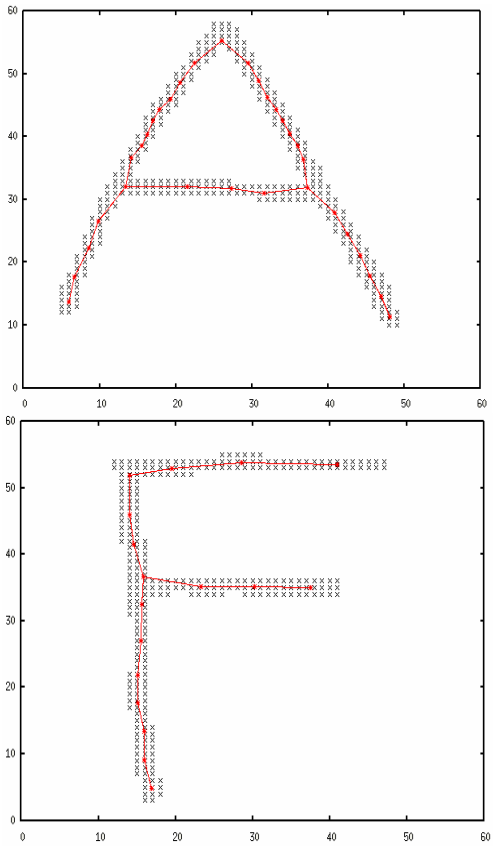


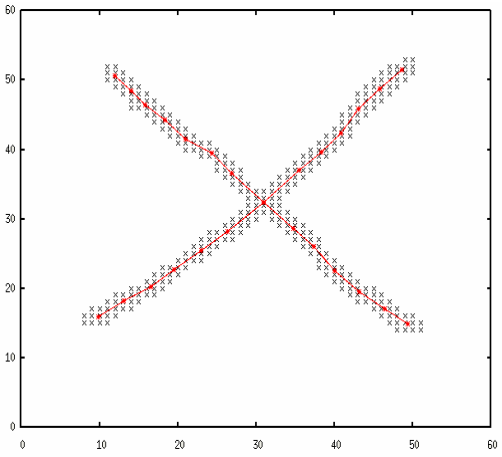
(a) 工具的測試



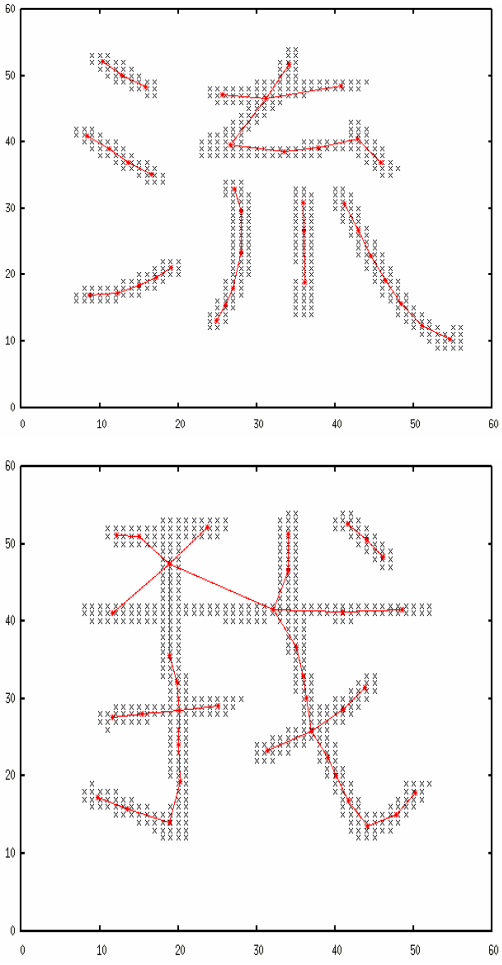


(b) 數字的測試





(c) 英文字母的測試



(d) 中文字的測試

圖 10 各種圖形的實驗結果

從表格 1 可以發現，本研究所提出的方法平均可以有效的提升 DSOM 網路 19%的速度。從圖 10 中則可以看出本研究對於大部份的圖形，均可以正確的抽出骨架。然而從數字「5」的骨架中可以發現，DSOM 網路仍會產生一些小分支；而從「我」字中更可以看出，

叉點變形的問題依然無法避免。這主要是因為在連結神經元的過程中，由於部份神經元具有過多相鄰的神經元以至於連結時產生錯誤。以「我」字為例，從圖 11 中可以發現此圖形共有三處以上的叉點變形(用圓形符號所標記的地方，共有三處)。其中 b 與 c 處利用後處理所提的方法可以得到適當的改善。然而，a 處由於 d 點剛好處在兩個變形區域中，使得此處變形的情況更為複雜。經由後處理之後，雖然變形的區域被新神經元所取代了。然而卻也造成了新神經元同時成了其它 5 個神經元的鄰近神經元，而使得此處連結出現錯誤的情況。而如何避免發生這種錯誤，即是本研究未來改進的目標。

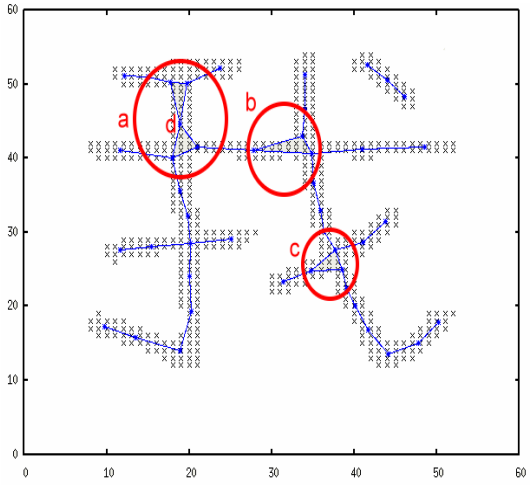


圖 11「我」字所產生的叉點變形

四、結論

本研究所提出的端點偵測及影像分割的方法，不僅可以擷取出與 DSOM 相同效果的骨架，其速度更比單純使用 DSOM 網路要快上 19%。然而本研究目前主要專注於速度上的改進，因而對於叉點變形及產生分支的問題還無法完全改善，而這也是本研究未來所要努力的目標。

五、參考文獻

[1] Q.Y. Ye and P.E. Danielsson, “Inspection of printed circuit boards by connectivity preserving shrinking,” IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.10, no.5,Sept. 1988,pp.737-742.

[2] L. Lam, S.W. Lee and C.Y. Suen, “Thinning methodologies - a comprehensive survey, ” IEEE Trans. on Pattern Analysis and Machine Itelligence, Vol .14 , no. 9, Sept. 1992, pp.869-885.

[3] T.Y. Zhang and C.Y. Suen, “A fast parallel algorithm for thinning digital pictures”, Communications of the ACM archive, Vol.27, no.3, Mar.1984, pp.236-239.

[4] Y.Y. Zhang and C.Y. Suen, “A modified parallel thinning algorithm,” IEEE 9th Int. Conf. on Pattern Recognition, Vol.2, Nov.1988, pp.1023-1025..

[5] P. Bhattacharya and X. Lu, “New parallel algorithm for thinning binary images,” in Proc. IEEE Int. Conf. on Systems, Conference Proceedings, vol.1, Oct.1991, pp.651-654.

[6] F.Y. Shin and W.T. Wong, “Fully Parallel Thinning with Tolerance to boundary noise,” Pattern Recognition, vol. 27, No.12, pp. 1677-1695, 1998.

[7] T. Kohonen, “The self-organizin map,”Proceedings of the IEEE, Vol.78, no.9, pp.1464-480,1990.

[8] B. Fritzke, “Let it grow – Self-organizing feature maps with problem dependent cell structure.” Artificial Neural Networks. North-Holland, Amsterdam, vol.1. pp.403-408.

[9] A. Datta and S.K. Parui, “Skeletons f from dot patterns: A neural network approach,” Pattern Recognition Letters, vol.18, pp.335-342, 1997.

[10] A. Datta, S.K. Parui and B.B. Chaudhuri, “Skeletal shape extraction from dot patterns by self-organizing,” in Proc. IEEE 13th Int. Conf. on Pattern Recognition, vol.4, Aug.1996, pp.80-84.