

Description :

一開始，先處理圖片檔的標頭資訊，包括 size, offset, width, height, bits 等資訊，之後再將圖片的色彩值讀入，因為這次範例圖片只有 8 位元色彩，所以每個畫素只需要 1byte 即可儲存。而這次作業是對圖片來處理 thinning，處理完圖片之後，再將標頭與畫素資料一起寫出並存，並且產生 thinning_lena 圖片，即可完成這次的作業。

Algorithm :

一開始的 binary，使用非常簡單的迭代演算法即可，將數值小於 128 全部調整為 0，大於 128 的則是調整為 255。

再者的 marked border operator，則是透過之前寫過的 Yokoi connectivity number 演算法，標記出所有的標籤，因為我們要找出 edge，所以我們考慮標籤為 1 的情況。

接著 pair relationship operator，則是透過與 4 連通的方式與周圍的像素標籤來做比較，如果有 1 出現的情況，我們就將它標記起來。

最後的 connected shrink operator，透過上面的輸出，我們會得到一個標記表，透過它並與原圖做比較，決定是否要刪除此點，透過上述的步驟不停迭代，直到沒有改變發生，我們就結束迭代，別判斷最後的二極圖如果為 1，就填入 255，反之就填入 0，產生圖片並輸出。

PrincipalCode :

◎Pair_Relationship

```
for(i=0; i<514; ++i)
    for(j=0; j<514; ++j)
        //檢查周圍是否有 1 出現，有就標記 p，否則標記 q
        if(1 == output[i][j] && (1 == output[i-1][j] || 1 == output[i+1][j] || 1 == output[i][j-1] || 1 == output[i][j+1] || 1 == output[i-1][j-1] || 1 == output[i+1][j+1] || 1 == output[i+1][j-1] || 1 == output[i-1][j+1]))
            pairmatrix[i][j]='p';
        else
```

```
pairmatrix[i][j]='q';
```

◎Connected_Shrink

```
for(i=0; i<514; ++i)
    for(j=0; j<514; ++j)
        //當像素被標記的時候 · 檢查周圍像素
        if('p' == pairmatrix[i][j])
            c = 0;
            check = true;
            //檢查是否為可以去除的點
            if((scale[i-1][j]) && (!scale[i-1][j-1] || !scale[i][j-1])) ++c;
            if((scale[i][j+1]) && (!scale[i-1][j+1] || !scale[i-1][j])) ++c;
            if((scale[i][j-1]) && (!scale[i+1][j-1] || !scale[i+1][j])) ++c;
            if((scale[i+1][j]) && (!scale[i+1][j+1] || !scale[i][j+1])) ++c;
            if(1 == c)
                scale[i][j] = 0;
```

◎Main

```
//直到圖片不再發生變化 · 跳出迴圈
while(check)
    check = false;
    Yokoi_Connectivity(output,scale);
    Pair_Relationship(output, pairmatrix);
    Connected_Shrink(pairmatrix, scale, &check);

//將結果輸出成圖片並存檔
for(int i=1; i<513; ++i)
    for(int j=1; j<513; ++j)
        if(scale[i][j])
            BMPdata[512-i][j].color = 255;
        else
            BMPdata[512-i][j].color = 0;
```

Parameters :

編譯程式碼 `g++ -o lena lena.cpp`

執行程式 `./lena lena.bmp`

lena.bmp 是我們的 Input Image

ResultingImages :

thinning_lena.bmp

