電機所-陳治言-R06921084

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。而這次作業是處理 Yokoi connectivity number。將所有的 label 處理完之後，輸出成文字檔儲存，產生出 Yokoi 的檔案，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 downsampling，則是將圖片以每 8 個點來進行取樣，最後將圖片降維成 64x64 的圖片。

最後的 Yokoi connectivity number 處理，使用上課教的的迭代演算法來做計算，並以 4 連通來做計算，得到每個點的 q 與 r 之後，將所有的 label 都完整標記上去。最後輸出的時候，將 label 為 0 的像素以空格來做輸出而其他像素則是以其 label 號碼做為輸出。

PrincipalCode：

◎Binary_Scale

```
//在取樣的過程中，順便做圖片二極化處理，並在圖片加上一層外框，以利後續處理。
int scale[66][66];
for(i=bmpInfo.biHeight-1; i>-1; i-=8)
    for(j=0; j<bmpInfo.biWidth; j+=8)
        scale[64-(i>>3)][1+(j>>3)] = (BMPdata[i][j].color < 128 ? 0 : 1);
```

◎Yokoi_Connectivity

```
//找出圖片中為 1 的地方進行處理，並初始 r、q 與 4 連通方向
for(i=0; i<66; ++i)
    for(j=0; j<66; ++j)
        if(scale[i][j])
            r = 0; q = 0; up = i>0 ? i-1 : 0; down = i<65 ? i+1 : 65;
            left = j>0 ? j-1 : 0; right = j<65 ? j+1 : 65;
```

```
//對 4 個方向進行處理，並計算 r 與 q 的數量
if(scale[up][j])
    if(scale[up][left] && scale[i][left]) ++r;
    else ++q;
if(scale[i][right])
    if(scale[up][right] && scale[up][j]) ++r;
    else ++q;
if(scale[i][left])
    if(scale[down][left] && scale[down][j]) ++r;
    else ++q;
if(scale[down][j])
    if(scale[down][right] && scale[down][j]) ++r;
    else ++q;

//處理完之後，發現 r 等於 4 的話，因為是 interior，所以設定為 5
if(r == 4)
    output[i][j] = 5;
//其餘的則是將 q 的數值當作 label 寫入
else
    output[i][j] = q;

//最後將所有的 label 輸出成檔案，並將 label 為 0 的以空格代替
FILE* fptr = fopen("Yokoi","w");
for(i=1; i<65; ++i)
    for(j=1; j<65; ++j)
        if(!output[i][j])
            fprintf(fptr, " ");
        else
            fprintf(fptr, "%d", output[i][j]);
        fprintf(fptr, "\n");
```

Parameters：

編譯程式碼 g++ -o lena lena.cpp

執行程式./lena lena.bmp

lena.bmp 是我們的 InputImage

ResultingImages：