電機碩 1-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 1 byte 即可儲存。而作業分為二部份，分別為 dilation 和 erosion。處理完所有的圖片之後，再將標頭與畫素資料一起寫出並存，並且分別產生 4 個檔案，dilation_lena、erosion_lena、opening_lena、closing_lena 四張輸出圖片，即可完成這次的作業。

Algorithm：

一開始的 dilation，我們將搜尋原圖中灰階值大於零部份，並將 kernel 上的參考點加上 value 之後附加到周圍，如果發現同樣位置有其他像素值，我們就取較大值為基準，就會形成淺色向外擴張的圖片。

再來的 erosion，則是透過比對 kernel 是否被包含於原圖中，如果為是，我們就將 kernel 中的每個像素值加上 value，取較小值為基準填入參考點中，如果為否，則填上黑色，則會形成黑色部分向外擴張的情況。

在 opening 與 closing 的部分，則是透過組合使用 dilation 和 erosion 的 方式，opening 是先使用 erosion，再來做 dilation，而 closing 則剛好相反。

Principal Code：

◎Dilation

```
for(i=bmpInfo.biHeight-1; i>-1; --i)
    for(j=0; j<bmpInfo.biWidth; ++j)
        //先確認該像素大於 0
        if(source[i][j].color)
            for(x=i+2; x>i-3; --x)
                for(y=j-2; y<j+3; ++y)
                    //不考慮的部分
                    if(x == i+2 && y == j-2 || x == i+2 && y == j+2 || x == i-2 && y == j-2 || x
                    == i-2 && y ==j+2)
                        continue;
```

```
            //如果沒有超出圖片邊界，將 kernel 給填上去，並取最大值
            if(x > -1 && x < bmpInfo.biHeight && y > -1 && y < bmpInfo.biWidth &&
            destination[x][y].color < source[i][j].color)
                destination[x][y].color = source[i][j].color;
```

◎Erosion

```
for(i=bmpInfo.biHeight-1; i>-1; --i)
    for(j=0; j<bmpInfo.biWidth; ++j)
        for(x=i+2; x>i-3; --x)
            for(y=j-2; y<j+3; ++y)
                //不考慮的部分
                if(x == i+2 && y == j-2 || x == i+2 && y == j+2 || x == i-2 && y == j-2 || x
                == i-2 && y == j+2)
                    continue;

                //超出邊界或是無法包含 kernel
                if(x < 0 || x >= bmpInfo.biHeight || y < 0 || y >= bmpInfo.biWidth
                || !source[x][y].color)
                    goto fail;

                //取 kernel 中的最小值並填入參考點中
                if(source[x][y].color < _min)
                    _min = source[x][y].color;
                destination[i][j].color = _min;
                fail:_min = 255;
```

◎Opening

```
//先使用 erosion，在執行 dilation
dilation(erosion(source));
```

◎Closing

```
//先使用 dilation，在執行 erosion
erosion (dilation (source));
```

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

Resulting Images：

dilation_lena.bmp

erosion_lena.bmp

opening_lena.bmp

closing_lena.bmp