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 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 5 個檔案，dilation_lena、erosion_lena、opening_lena、closing_lena、hit_and_miss_lena 五張輸出圖片，即可完成這次的作業。

Algorithm :

一開始的 dilation，我們將搜尋原圖中的白色部份，並將 kernel 的白色像素給附加上去，就會形成白色向外擴張的圖片。

再來的 erosion，則是透過比對 kernel 是否被包含於原圖中，如果為是，則將白色像素給附加上去，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。

最後的 hit and miss，則是將原圖和其補圖與二種不同 kernel 作 erosion 之後，再將得到的二張圖做交集運算，最後就會得到類似輪廓的圖形。

Principal Code :

◎**Dilation**

```
for(i=bmpInfo.biHeight-1; i>-1; --i)
   for(j=0; j<bmpInfo.biWidth; ++j)
      //先確認該像素為白色
      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
    if(x > -1 && x < bmpInfo.biHeight && y > -1 && y < bmpInfo.biWidth)
        destination[x][y].color = 255;
```

◎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;

                    //因為超出邊界，故視為無法包含 kernel
                    if(x < 0 || x >= bmpInfo.biHeight || y < 0 || y >= bmpInfo.biWidth)
                        goto fail;

                    //原圖無法包含 kernel
                    if(!source[x][y].color)
                        goto fail;

                    //原圖包含 kernel，故將白色給填入
                    destination[i][j].color = 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));
```

◎Hit and Miss

```
//對其原圖做 erosion
BMPoutput = erosion (BMPdata);
```

```
//取其補圖
for(i=0; i<bmpInfo.biHeight; ++i)
    for(j=0; j<bmpInfo.biWidth; ++j)
        BMPdata[i][j].color = BMPdata[i][j].color ^ 255;

//對補圖做 erosion
BMPoutput2 = erosion (BMPdata);

//將二者結果取交集運算
for(i=0; i<bmpInfo.biHeight; ++i)
    for(j=0; j<bmpInfo.biWidth; ++j)
        BMPdata[i][j].color = BMPoutput[i][j].color & BMPoutput2[i][j].color;
```

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

hit_and_miss_lena.bmp