

電腦視覺作業五

Computer Vision HW5




R04525092 工科碩二 鄭力文

<https://github.com/Mike-Zheng/NTU-Computer-Vision-I>

使用語言 C++ with openCV

Write programs which do gray scale morphological dilation, erosion, opening, and closing on a gray scale image

Kernel use:

Kernel																																																			
<table><tr><td></td><td>*</td><td>*</td><td>*</td><td></td></tr><tr><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td></tr><tr><td>*</td><td>*</td><td></td><td>*</td><td>*</td></tr><tr><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td></tr><tr><td></td><td>*</td><td>*</td><td>*</td><td></td></tr></table>		*	*	*		*	*	*	*	*	*	*		*	*	*	*	*	*	*		*	*	*		<p>Please use the octonal 3-5-5-5-3 kernel with value = 0</p> <p>(which is actually taking the local maxima or local minima respectively).</p> <table><tr><td></td><td>0</td><td>0</td><td>0</td><td></td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td></td><td>0</td><td>0</td><td>0</td><td></td></tr></table>		0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	
	*	*	*																																																
*	*	*	*	*																																															
*	*		*	*																																															
*	*	*	*	*																																															
	*	*	*																																																
	0	0	0																																																
0	0	0	0	0																																															
0	0	0	0	0																																															
0	0	0	0	0																																															
	0	0	0																																																

1. Dilation

膨脹將每一點透過 kernel 進行運算，該點中心吻合 kernel 的周邊為 255。

//進行膨脹

```
int dilation(int x,int y,Mat im){

    int localMax=0;
    for(int j=-2;j<2;j++){
        for(int i=-2;i<2;i++){
            if(kernel[i+2][j+2]==255){
                if(localMax < im.at<Vec3b>(x+i,y+j)[0])
                    localMax = im.at<Vec3b>(x+i,y+j)[0];
            }
        }
    }
    return localMax;
}
```



2. Erosion

//進行消退

```
int erosion(int x,int y,Mat im){
    int localMin=255;
    if(x>1&& y>1)
        for(int j=-2;j<2;j++){
            for(int i=-2;i<2;i++){
                if(kernel[i+2][j+2]==255){
                    if(localMin > im.at<Vec3b>(x+i,y+j)[0])
                        localMin = im.at<Vec3b>(x+i,y+j)[0];
                }
            }
        }
    return localMin;
}
```



3. Opening

先消退再膨脹



4. Closing

先膨脹再消退

