數位視訊技術

105323031蔣圳嵐

# 原理:

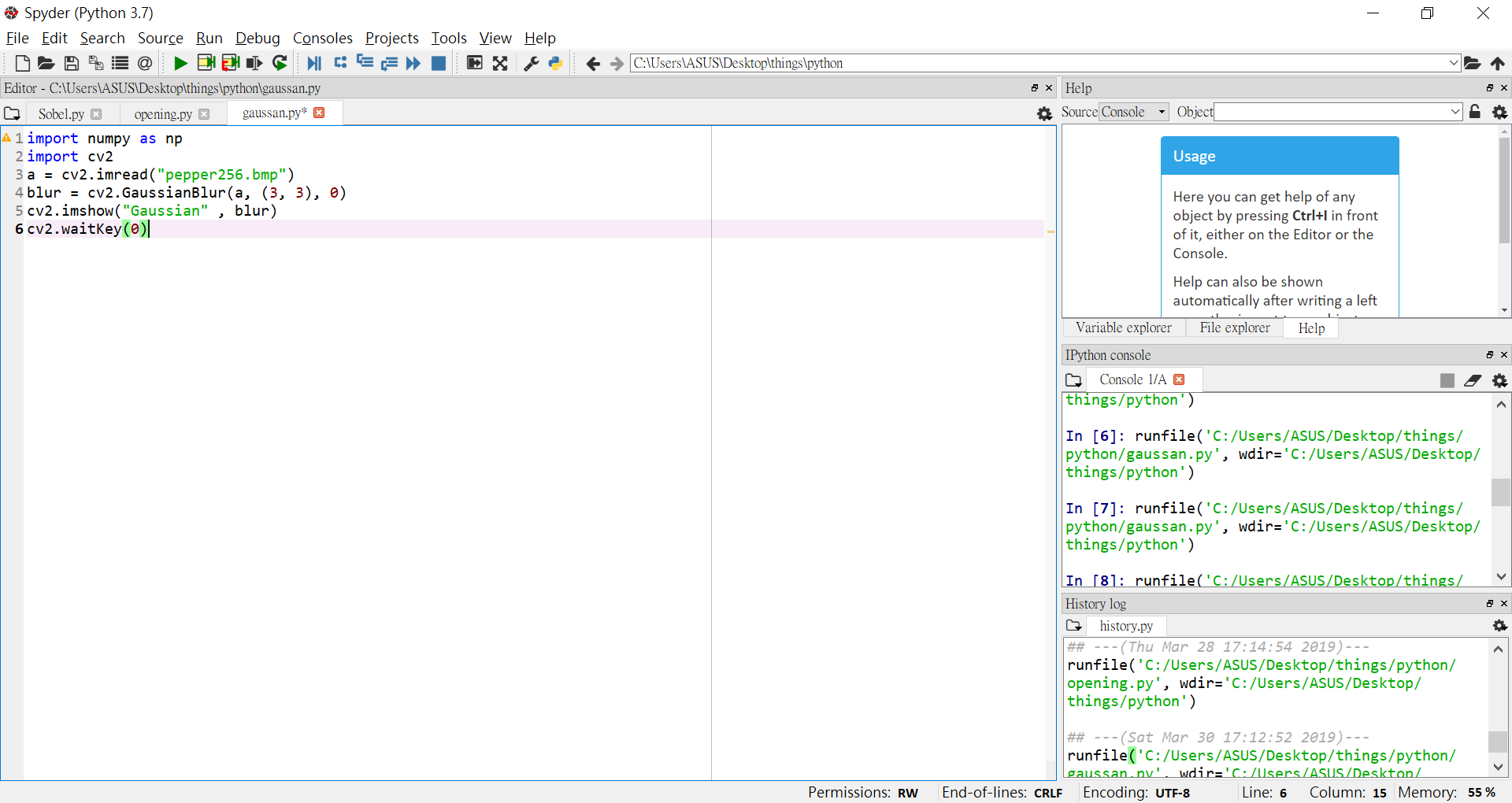
Guassian smooth:利用guassian distribution做出一個mask，再拿去和原圖像convolution，就可以得到一個較模糊的照片，可以將圖像中的一些雜訊濾掉。

Edge detection:我是利用Sobel的方式，也同樣是把一個矩陣拿去和原圖像convolution，而我會選擇Sobel的原因是，Sobel縱橫的方向都可以同時取得。

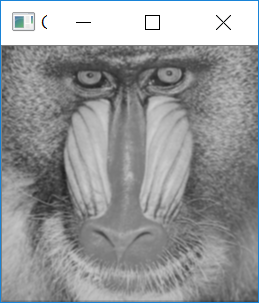
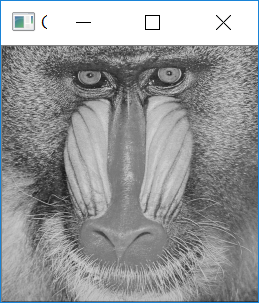
Opening and Closeing:首先要先明白Dilation(擴張)和erosion(侵蝕)，顧名思義前者會把原圖像變的肥胖，而後者則是淘汰掉小於輸入矩陣大小的影像，因此叫侵蝕。而Opening為先做erosion再做dilation，Closing為先做dilation再做erosion。

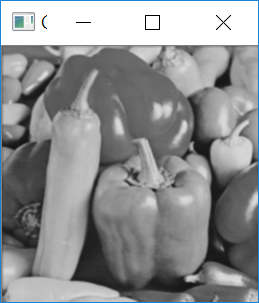
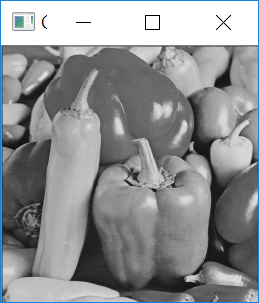
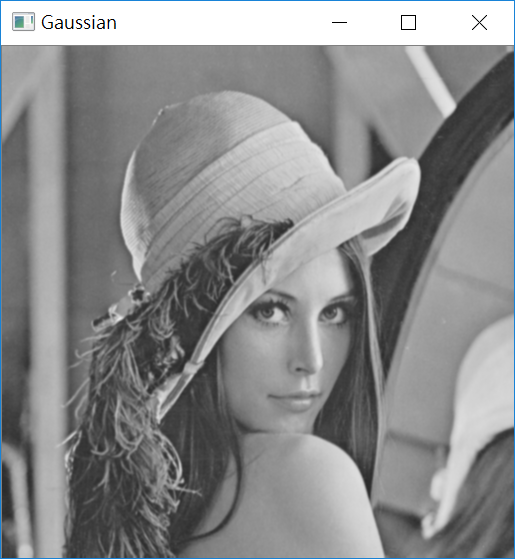
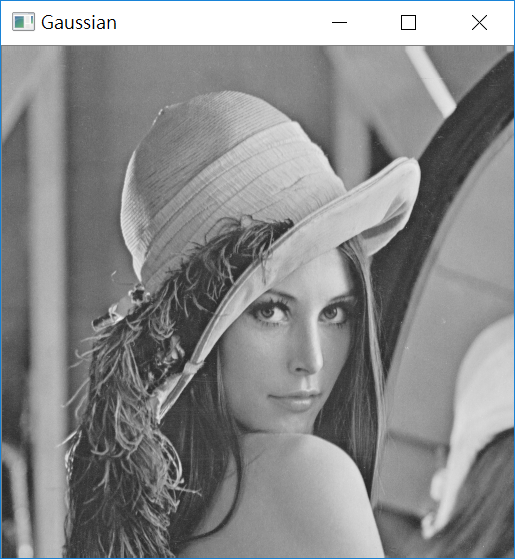
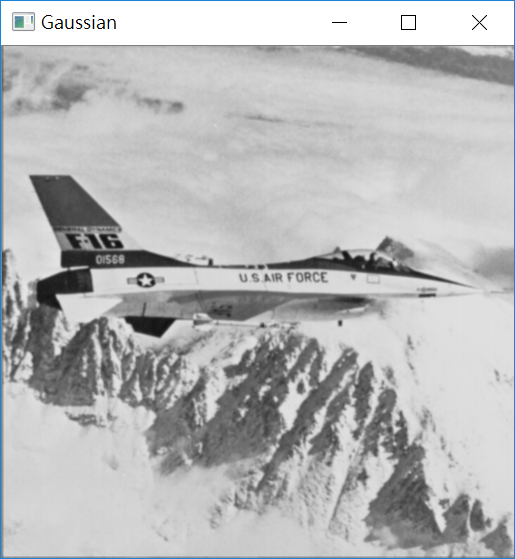
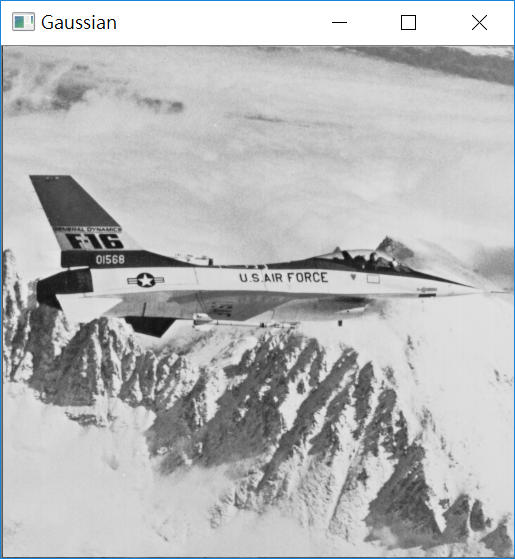
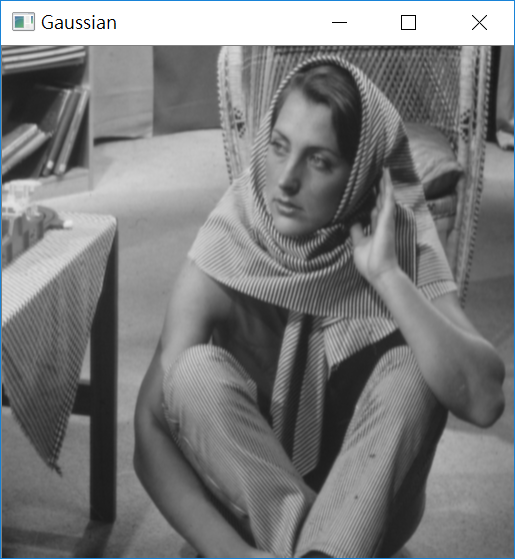
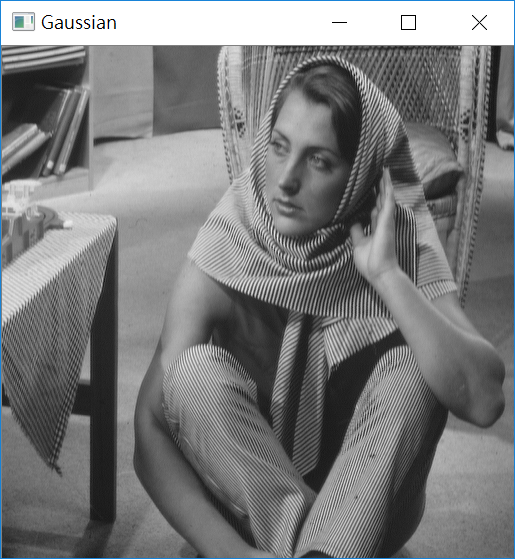
# 程式碼和結果:

Guassian Smooth:

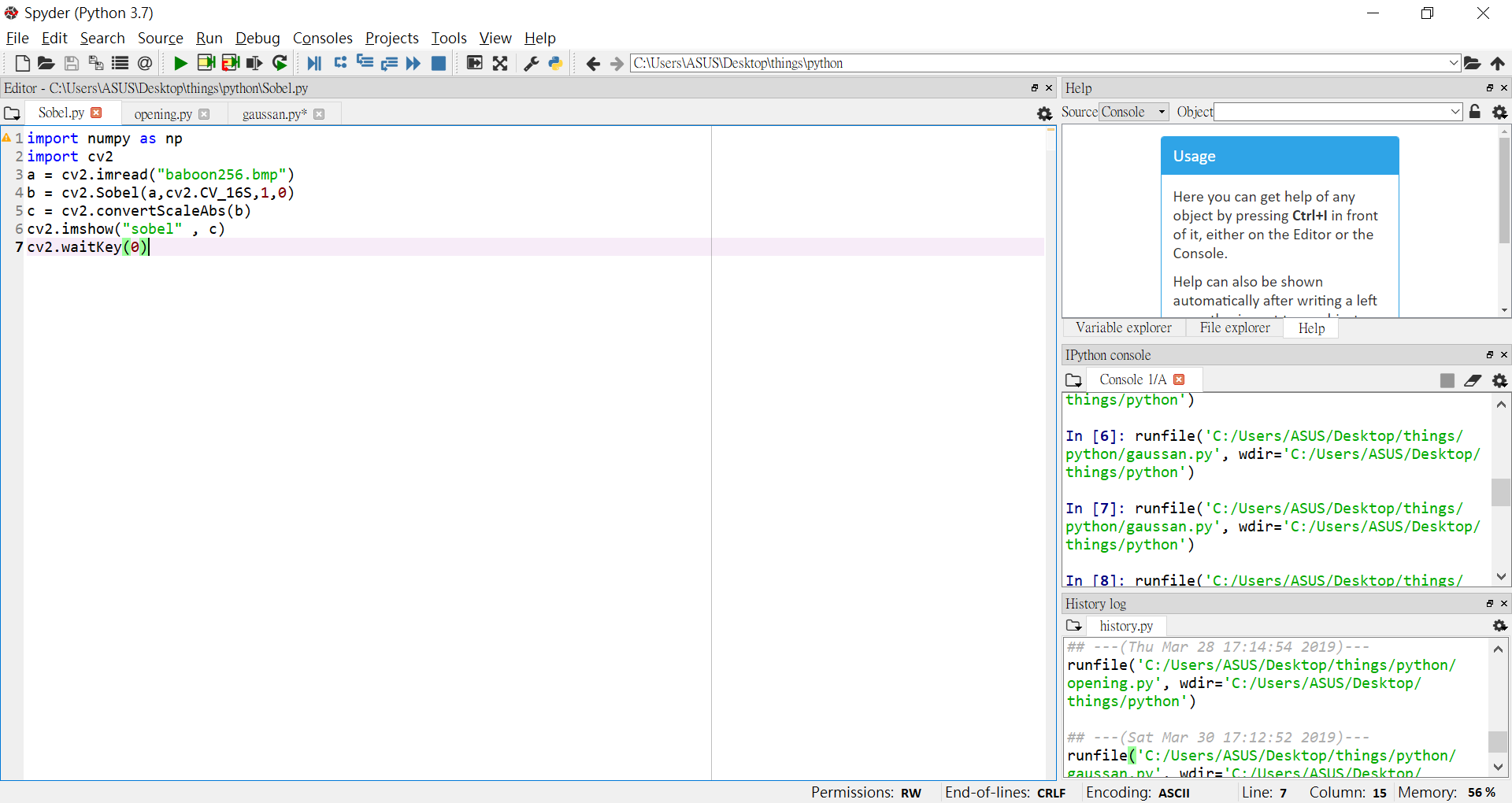


利用opencv內建的一個函式GaussianBlur就是gussian distribution的一個mask，後面的參數分別為輸入圖像、mask大小、顏色標準差，顏色標準差設為0則是讓程式自己運算合適的值，因此我設定為0。

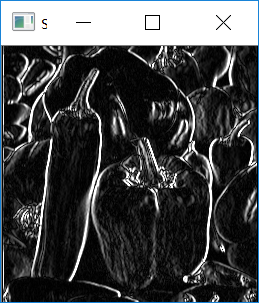
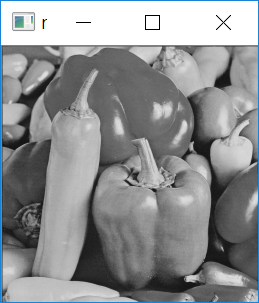
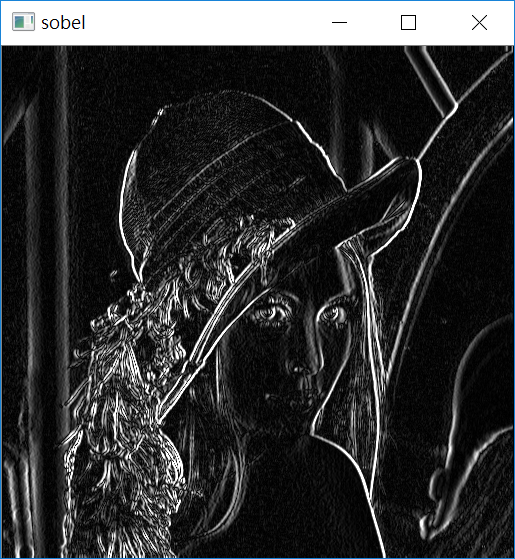
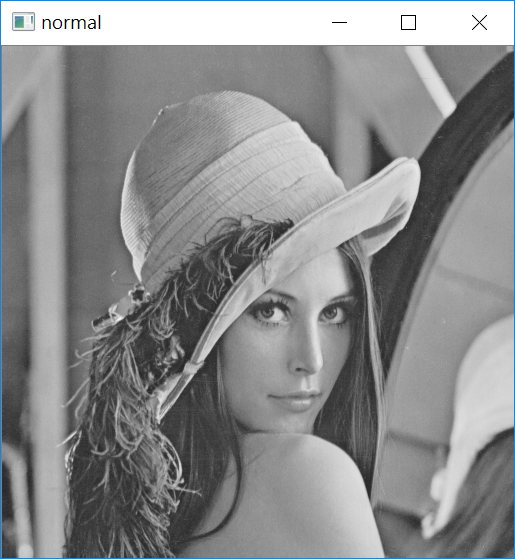
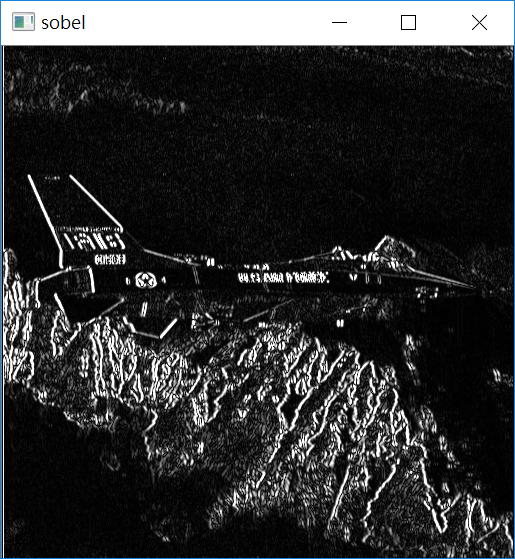
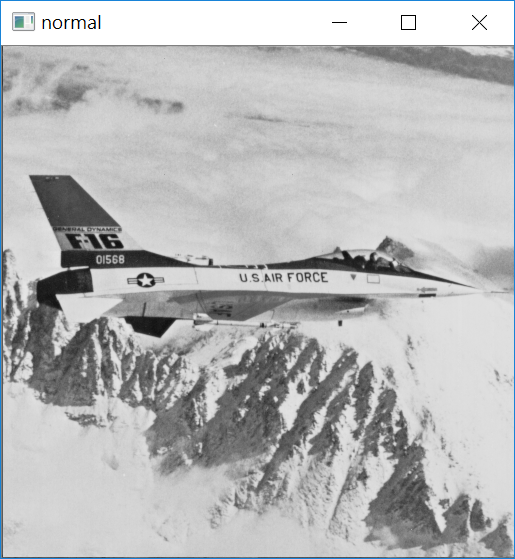
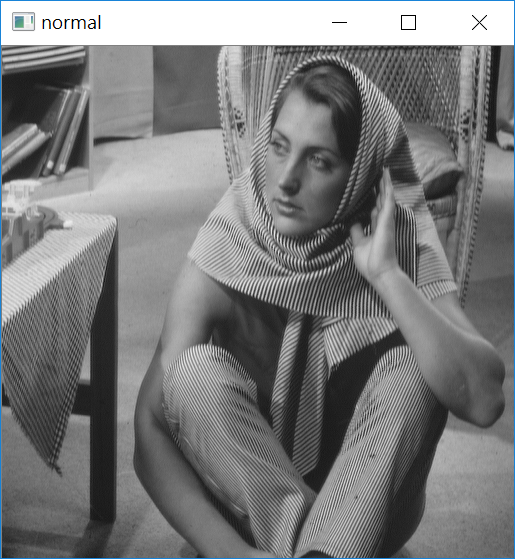
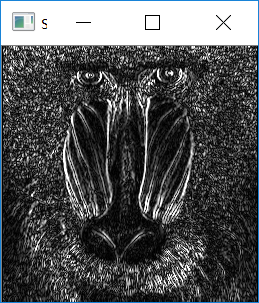
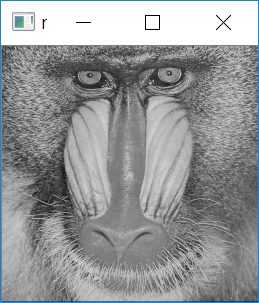




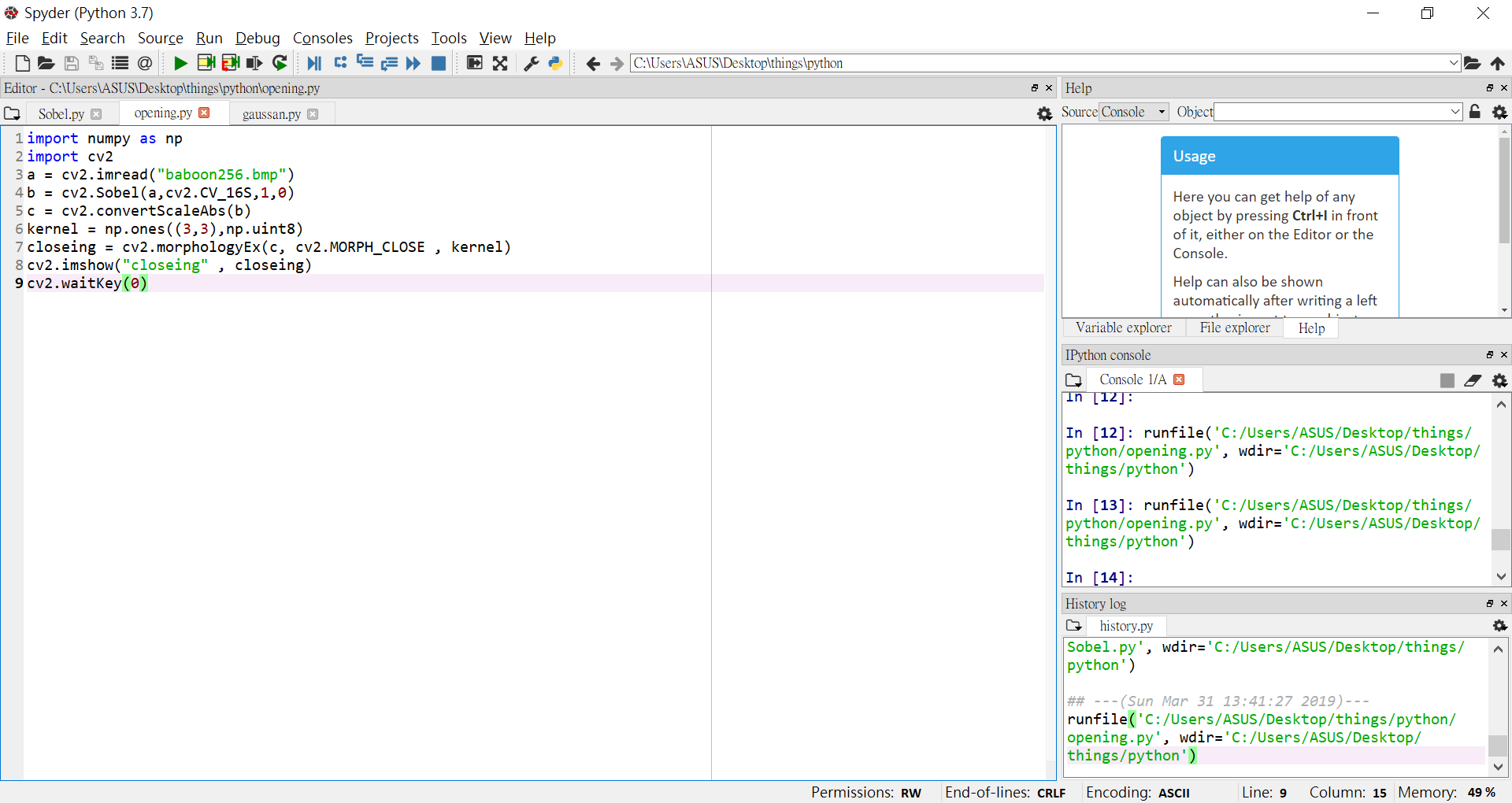
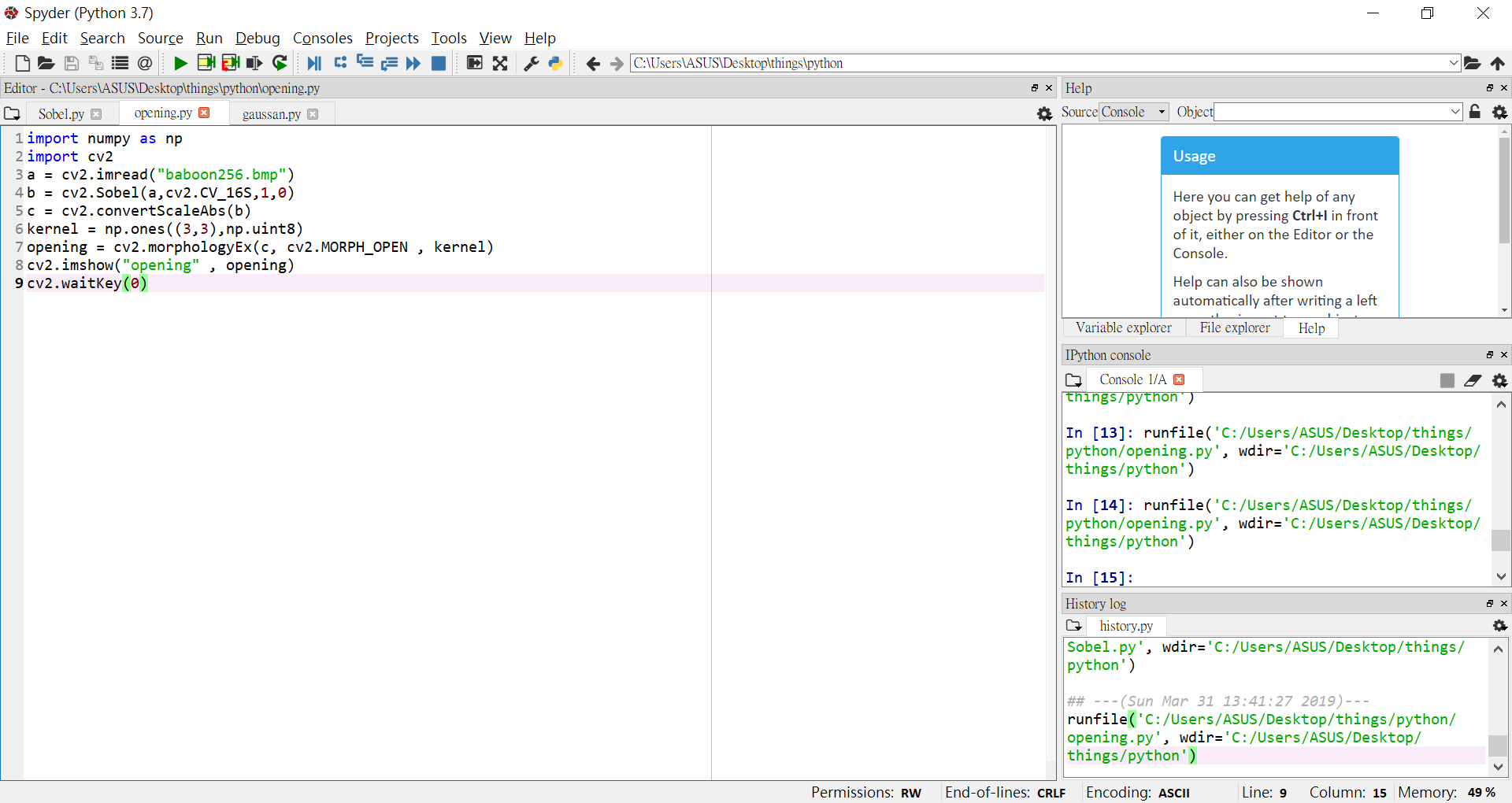
Edge detection:



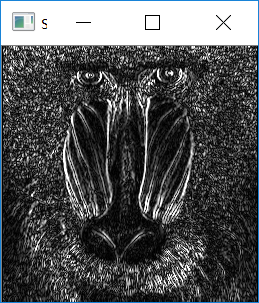
邊緣偵測我使用了Sobel這個方法，而指令則是使用cv2.Sobel，他有四個參數分別為:輸入圖片、圖像深度、dx,dy的求導皆數。第二個參數我使照片轉為16位元，因此最後要再使用cv2.convertScaleAbs這個函式將輸入圖像轉變成8位元，最後才能正常顯示。

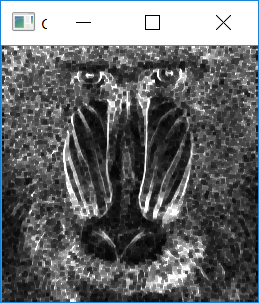
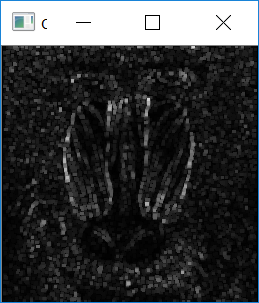


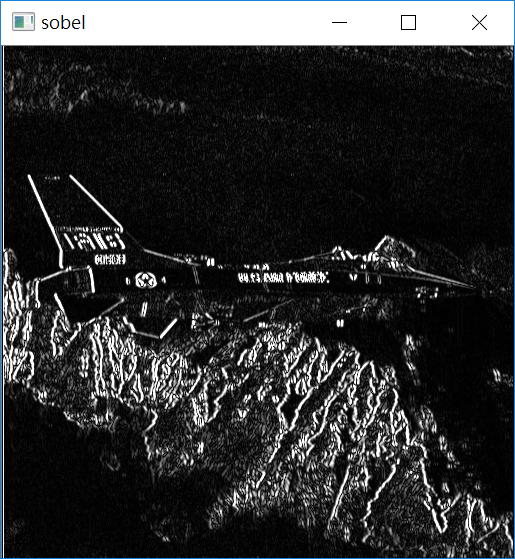
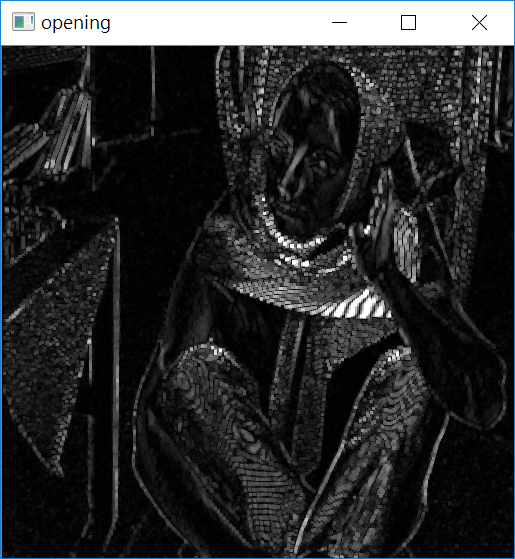
Opening and Closeing:

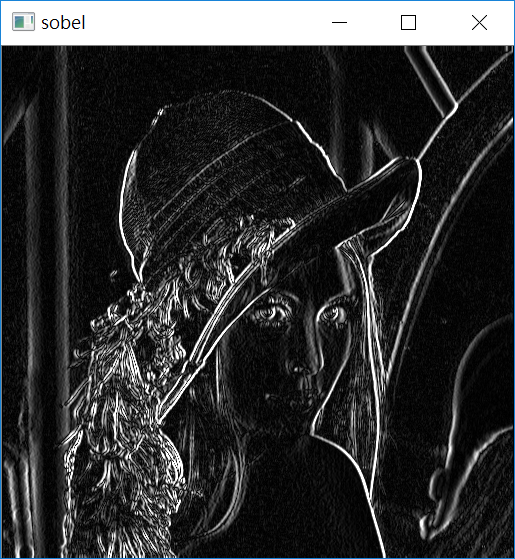


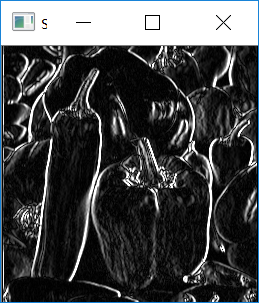
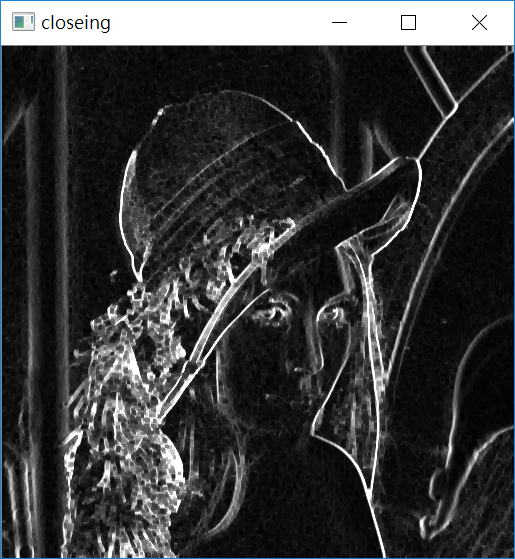
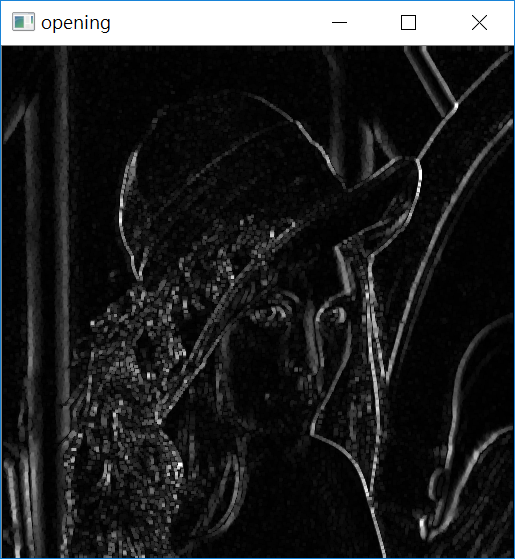
opening和closeing則是拿前面做好Sobel的圖像再宣告一個3x3的矩陣叫做kernel，然後再使用cv2.morphologyEx的函式，他也一樣有三個參數:輸入圖像、要做OPEN還是CLOSE、矩陣大小。

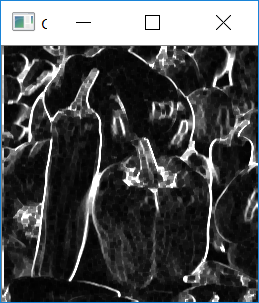
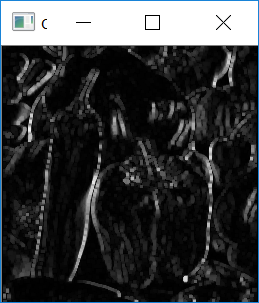












# 結論:

Guassian smooth:因為我沒有在圖像中加入雜訊，因此Guassian smooth的效果沒那麼顯著，只是單純把圖像做模糊的狀況。但這個也告訴我們世界上沒有一個事物是完美的，要濾雜訊就必須犧牲圖像的清晰度，是一個雙面刃。

Edge detection:邊緣偵測是我認為做最有感的一個，因為之前我總是很好奇邊緣到底是如何找到的，沒想到只要設計好矩陣再拿去和圖像convolution就可以得到了。這樣以後可以拿這些處理過後的圖片來train就可以讓神經網路學習我們想要他們注意的地方。

Opening and Closeing:Opening的效果不是很顯著，通常都會讓影像的邊緣變得更糊，我想這個的應用應該是把影像內部一些小小的邊緣都模糊掉，只留下邊緣較粗的紋路。而Closeing就是把整個的邊緣紋路再更加深，讓整個效果更明顯更好辨識。