

HW6-110810006林君曆

Canny Edge Detection

1. Noise Reduction

- a. Use the Gaussian filter with a (kernel size=5, sigma=1.1) kernel

延續使用HW5的Gaussian filtering function，不過在邊緣一圈的pixel保留原本圖像的pixel，不進行filter

- func gaussian_kernel
 1. 接收kernel size和sigma的參數
 2. 計算kernel兩側寬度， $k = \text{size} // 2$
 3. 初始kenel陣列
 4. 遍歷kernel每個座標，計算gaussian值
 5. normalize kernel值
- func gaussian_filter
 1. 使用gaussian_kernel函式建立gaussian kernel
 2. 計算pad_size用於後續slice region用、初始化filtered陣列
 3. 遍歷照片的每個像素點
 - a. 首先，slice以該點為中心的region
 - b. 將該region與gaussian kernel做卷積
 - c. 將卷積結果assign給filtered_image陣列

2. Find the intensity gradient(sobel operator) of the image

Sobel

-1	0	1
-2	0	2
-1	0	1

G_x

-1	-2	-1
0	0	0
1	2	1

G_y

$$G = \sqrt{G_x^2 + G_y^2}$$
$$\theta = \arctan\left(\frac{G_y}{G_x}\right)$$

func sobel_operator

1. 定義好sobel的兩個kernel、初始化Gx和Gy的陣列
 2. 遍歷所有pixel，將3*3的region與兩個kernel做捲積
 3. 將得到的Gx, Gy根據公式得到G，使用np.sqrt直接對矩陣計算
 4. 使用np.arctan2計算斜率角度(回傳範圍為弧度[-pi, pi])
- ### 3. Non-maximum suppression
- a. 使用np.rad2deg將弧度轉成角度
 - b. 將負角+180以限縮角度範圍為0到180之間，方便suppression的檢驗

- c. 遍歷所有pixel，根據該角度所屬四個方向中的哪一個，去看該像素是否有大於方向上兩側的neighbor，如果有就留下該像素的gradient，否則就設為0

4. Double threshold

根據Gradient的強度以及給予的low, high threshold劃分為weak, strong edge，其餘的則捨棄

5. Edge Tracking by Hysteresis

- a. 先宣告一個boolean陣列紀錄每個點是否被visit過
- b. 定義一個從strong edge開始spread的dfs function
 - i. stack初始只會有該strong edge
 - ii. 當stack不無空時，會不斷pop出點，並檢驗是否有在檢驗的範圍內(not visit, in boundary)
 - iii. 如果該點符合檢驗範圍，也是weak edge的話就設置為strong edge且設為visit，同時append到stack裡面繼續link更多的weak edge
- c. 遍歷所有edge陣列中的點，如果是strong就呼叫dfs function不斷的link其他的weak edge
- d. 再遍歷一次所有edge，將沒有被連到的weak edge歸零(去除false edge)

Result Images

- 經果幾次實驗調整後，最剛好的threshold是60和170，所呈現的效果最好
- 就我設定的threshold來看，第二張大象的圖片皺褶的部分比較沒有顯示出來，其餘兩張照片都非常好，也是我覺得很剛好的閾值
- 每一步驟的處理都略過圖像最外圍的一圈pixel

一開始我延續上次作業將gaussian filter做 zero padding再filtering，而其他的步驟因為題目沒有說，我就忽略了圖像外圍的點，但是後來發現這樣邊匡會全部一圈都被認作edge，一步步看每一步驟的結果後，最後把gaussian filter zero padding的部分不做後，結果就好很多



