DIP HW4　R06944023 吳尚真

PROBLEM 1: HOUGH TRANSFORM

a) 做 edge detection，輸出 E

1) Your motivation and approach:

考量到原圖是一個建築物，裡面的 edge 比較多是水平與垂直，因此我選擇使用 Lecture 03 教的 1st order edge detection 裡的計算 3 點求 gradient 的方法，因為這個方法適用於水平與垂直的 edge。

- **Discrete case**
  - **Approximation I – 3 points**
    - **Row gradient**

$$\frac{\partial F}{\partial x}(j,k) \cong F(j,k) - F(j,k-1) = G_R(j,k)$$

    - **Column gradient**

$$\frac{\partial F}{\partial y}(j,k) \cong F(j,k) - F(j+1,k) = G_C(j,k)$$

A_0	A_1	A_2
A_7	$F(j,k)$	A_3
A_6	A_5	A_4

$$\Longrightarrow G(j,k) = \sqrt{G_R^2(j,k) + G_C^2(j,k)} \qquad \theta(j,k) = \tan^{-1}\left(\frac{G_C(j,k)}{G_R(j,k)}\right)$$

至於這個方法，我在作業 2 就已經實做過，所以直接拿當時的 code 來用。

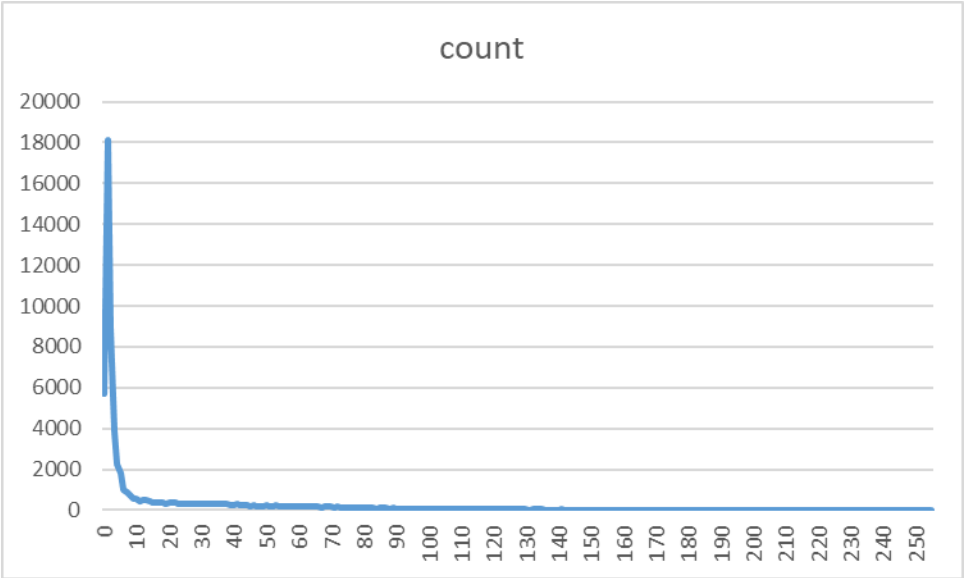
具體步驟如下:

- 先算出 row 跟 column 的 gradients。 這裡同時也一起計算 theta，以便下一題使用。
- 得到 gradient 後，計算相對應的 histogram 並繪出。
- 根據 histogram 決定 Threshold，小於 T 的地方設為 0，其他設為 1。

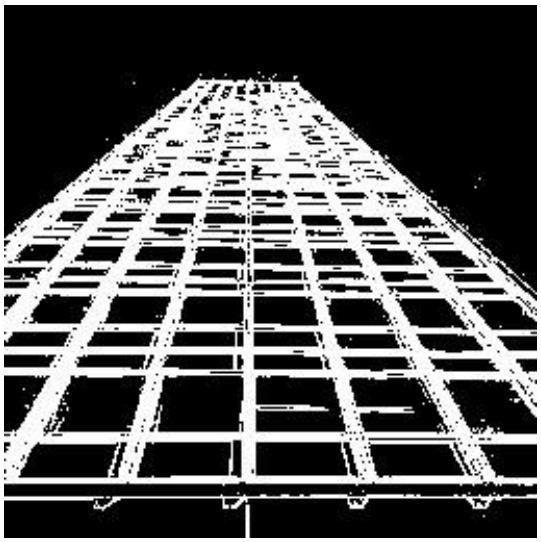
2) Original images

3) Output images

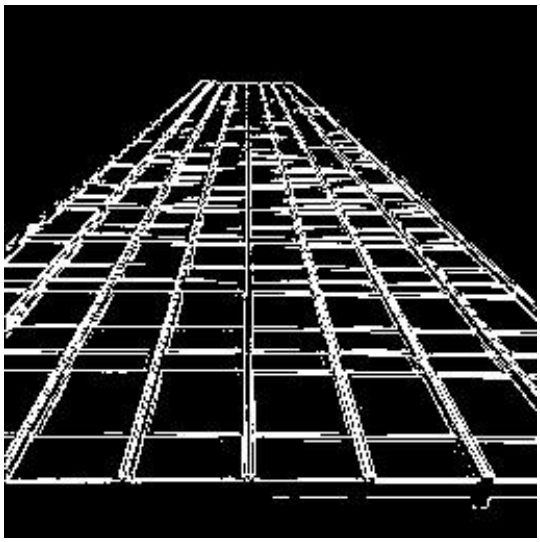
畫出 histogram 以利決定 Threshold:



T = 10



T=40



4) Discussion of results:

T 取的越大，得到的 edge 會越少。 一開始，我 T 是取 10，但是太多 edge 會讓(b)小題計算 rho, theta 的 count 數太大，(超過 255)，進而造成(b)與(c)小題的結果沒有差異，所以我將 T 取大一點，減少 edge。

由輸出的結果圖可以觀察到，圖片下面的 edge 因為在原圖中比較清晰(比較大，而且 intensity 的變化也較明顯)，所以 edge 偵測的結果還不錯；但是在圖片上方，因為方格(猜測應該是建築物的窗戶)較小，intensity 變化也較小，所以 edge 的偵測較差(有些方格都糊在一起)。

b) 對上一步的 E 做 Hough Transform，並將累積的 array 輸出成H_1。X 軸代表 theta，Y 軸代表 rho。

1) Your motivation and approach:

根據請教助教的結果，Hough Transform 的步驟如下:

- 先設定好 theta 會在 0-180 度之間變化，所以 accumulated array 的 X 軸就是從 0-180；而 rho 的範圍也可以事先算出，他的 upper bound 應該是 $\pm 128\sqrt{2} \approx \pm 181$，所以也可以訂出 accumulated array 的 Y 軸就是-181~+181。因此可以將 accumulated array 定義成 363X181 的 array。
- 藉由上一步求得的 edge map，對每一個 edge 去 iterate theta (0-180, increment = 1)，再根據下面的公式求 rho

$$\rho = (x - 128) \times \cos\theta + (y - 128) \times \sin\theta$$

- 有了計算出來的 rho 及指定的 theta 後，就可以在對應到的 accumulated array +1
- 每一個 edge 都 iterate 完畢後，就可以把 accumulated array 輸出出來

3) Output images

為了讓圖片有較明顯的對比，我會在 (c)小題中將兩個放在一起做對比

c) 對H_1做 contrast adjustment，輸出H_2

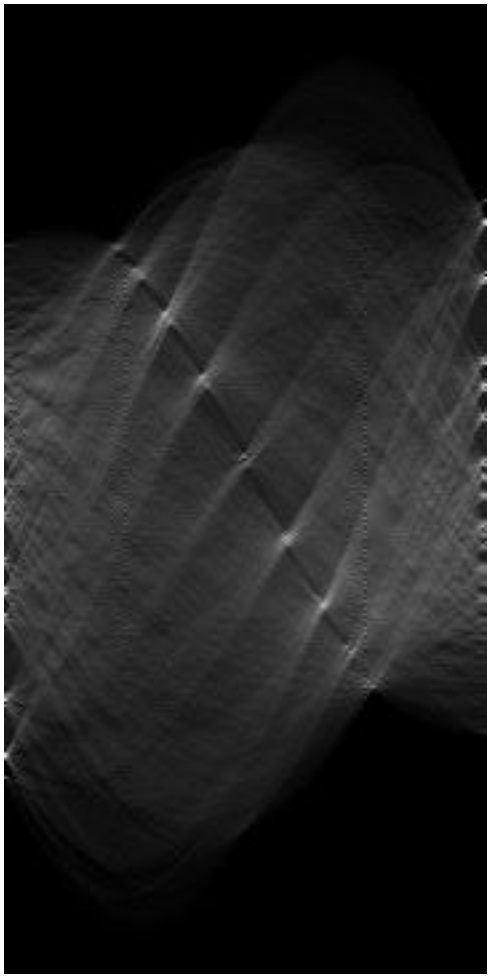
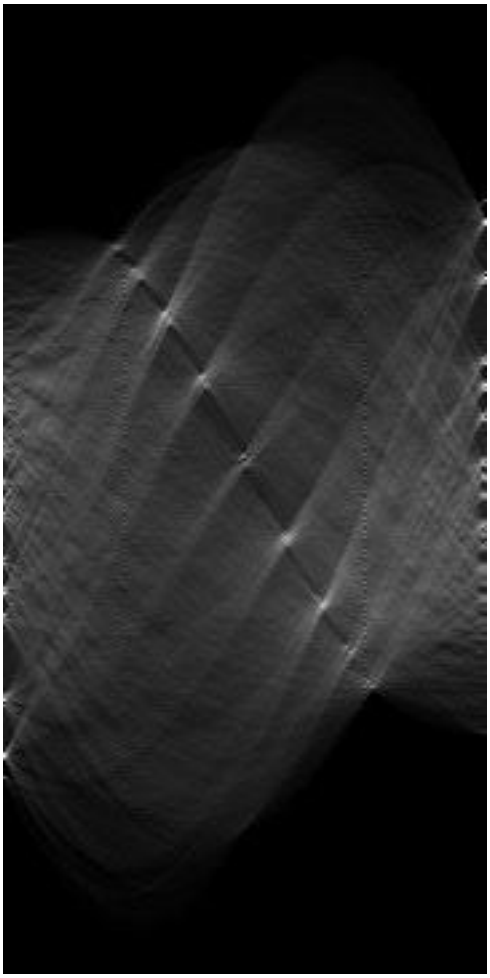
1) Your motivation and approach:

這裡我的做法就是將 accumulated array 每一格的 pixel value 做 normalization，步驟如下:

- 先 iterate through 整個 accumulated array，找出最大的 intensity value
- 再遍歷一次 accumulated array，此時將 accumulated array 裡面的值除以 maximum intensity value 再乘以 255，並將新計算出來的 value assign 給 adjust accumulated array
- 將 adjust accumulated array 輸出成H_2

3) Output images

左為 H_1。　右為H_2

4) Discussion of results:

再 normalize 每一個位置的 pixel value 後，可以些許的觀察到H_2在中間的 intensity 有比H_1提亮了一點，如果將兩張圖前後播放比對，會比較明顯。

d) 在 E 上畫出 10 條最明顯的線，存成D_1；在 E 上畫出 20 條最明顯的線，存成D_2

1) Your motivation and approach:

- 先 iterate through 上一題求到的 accumulated array，找出有最大 count 數的 rho 與 theta
- 根據 theta 值，反推兩點座標 (x1, y1)、(x2, y2)。
- 如果 theta 等於 0 或 180 度，表示應該是水平線，就設 y1 = 0, y2 = 256，利用下列公式，求 x1, x2

$$\mathrm{x} = \frac{\rho - (y - 128) \times \sin\theta}{\cos\theta} + 128$$

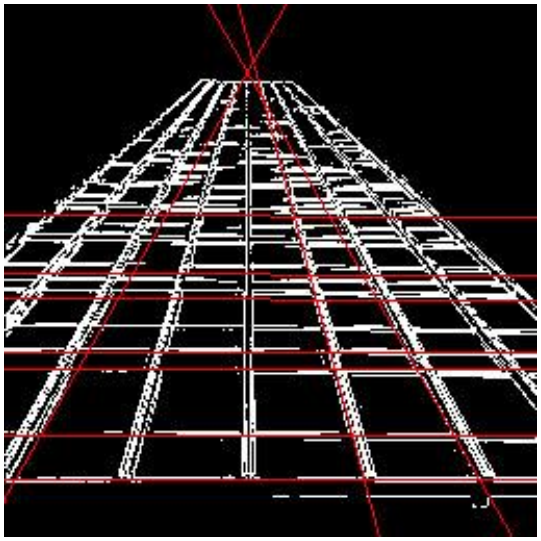
- 反之，則設 x1 = 0, x2 = 256，利用下列公式，求 y1, y2

$$\mathrm{y} = \frac{\rho - (x - 128) \times \cos\theta}{\sin\theta} + 128$$

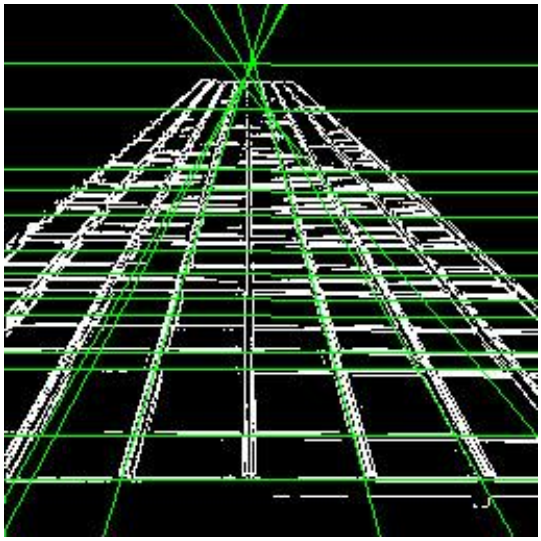
- 有了兩點座標後，就可以用兩點座標求斜率，畫出直線。值得注意的是，如果這條線的角度介於 45-135 或是介於 225-315，從 y 軸慢慢加值畫直線會比較好；反之，則從 x 軸畫直線會比較好。
- 因為要畫出不同顏色，所以先 copy 原來只有一個 channel 的 edge map，改為三個 channel 的 edge map，然後再利用與 HW3 相同的方法，對不同 channel assign 0 或 255，來改變顏色
- 為了避免畫出來的線太過靠近，所以每畫完一條線，我會將其鄰近區域的 rho 與 theta 都設為 0。 如果 theta = 0 的話，要同時將 theta = 180 的地方設為 0；如果 theta = 180 的話，也是相同的道理。

3) Output images

D1.raw (10 條線)



D2.raw (20 條線)



4) Discussion of results:

優點: 由上面兩張圖可以觀察到，對於 building 中的直線，單純使用 edge detection 可能會造成一些不連續，但是 Hough Transform 可以將不連續，但是應當要在同一條直線上的線連起來。

缺點: 因為 Hough Transform 會直接畫一條直線，所以對於 building 的頂端，其實那邊已經不是 edge 了，但線還是會穿過。

其實還是要看原圖的情況，如果原圖中的 object 有部分被擋住，這個特點就能將被擋住的 object 找出來；但是如果是像這次拿到的 building，並沒有東西被擋住，就會造成上方明明已經沒有 object 了，但是線還是會畫出來。