**Computer Vision Homework 2** B07902078 資工三 沈韋辰

Language: Python3.8.3

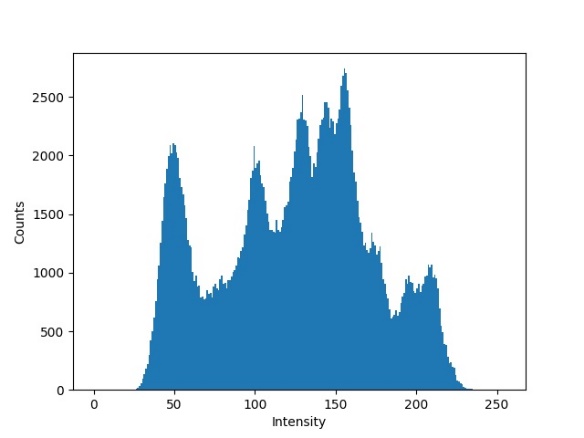
Modules: OpenCV(cv2), numPy, matplotlib.pyplot

1. Generate a binary image (threshold at 128):

**一張含有 文字 的圖片

描述是以非常高的可信度產生Algorithm:** Traversing all the pixel. If its intensity is higher than or equal to 128, then raise it to 255 (brightest); otherwise, reduce it to 0 (darkest).

1. Generate a histogram:

**Algorithm:** Accumulating the pixels with the same intensity and draw it by the module *matplotlib.pyplot.bar*

1. Find the connected components with centroid and bounding box.

In my method, I divide this problem to four steps:

**Step.1**

We need to turn lena.bmp into binary image. Just take the result of problem (a), let the intensity 255 be ‘1’ and 0 be ‘0’.

**Step.2**

In this homework, I adopt an algorithm which is similar to **flooding algorithm** with **4-connected** because I think the algorithms taught in class are not such intuitive and a little complicated. This method traverses all the pixels, if it hasn’t been labeled, then we put a barrel of “water” (label) on it and it will flood to 4 directions of its neighbors. If the neighbor has been labeled, the flood stops there; otherwise, we label the neighbor and continue the flood. We use a queue to maintain whom is the next one to be labeled and another list to record if the scale of this flood is over 500 pixels. This algorithm takes **O(RC)** time. (for size of image is RxC.) Each pixel just needs to check status, check neighbor status and labeled, so I think it is still a good way just like the algorithms taught in the lecture but easier to realize.

一張含有 文字, 光 的圖片

描述是以非常高的可信度產生 **Step.3**

After finishing labeling, we traverse the image again for measuring the boundary and centroid of each component. By the way, I also color a demo image to make sure if the algorithm worked.

**Step.4**

一張含有 文字 的圖片

描述是以非常高的可信度產生Finally, draw the boundary boxes and centroids by *cv2.rectangle and cv2.circle*.

**Bonus(?)**

These problems are found in the last page of slide:

* 舉兩種connected component labeling ，以及他們的傳播方向、次數與其特色。

1. Iterative Algorithm:
   * + 方向: 先**由上往下，由左往右**，再以相反方向再做一次
     + 次數: 基於圖形的圈數等因素，次數可能會多於一次
     + 特色: 由於要做到不再發生變化的平衡點才能停，因此時間複雜度很不穩定；但因撰寫容易，適合處理尺寸較小的圖片。
2. Classical Algorithm:

* 方向: 由上往下，由左往右
* 次數: 兩次，第一次做粗略的標籤和建立等價表，第二次再將相連的區塊合併。
* 特色: 只需遍歷兩次，因此時間複雜度穩定；但需要建立等價表，在做較大尺寸的圖片時會耗費較大的記憶體。
* Signature segmentation 可以有什麼切法，舉兩種。

1. Vertical and horizontal projections
2. Diagonal projections

* 舉出兩種你所知道的thresholding 演算法

1. Otsu’s threshold
2. Kullback-Leibler divergence threshold