Problem 1: MORPHOLOGICAL PROCESSING


sample1.png

(a) Design a morphological processing to extract the objects' boundaries in sample1.png


result1.png

Approach: 以 Dilation 與 Erosion 為基礎 Morphology。將圖片與自身 Erosion 的差為 Boundary Extraction，即:

$$B(Image) = Image - Erosion(Image, \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix})$$

Implementation:

- Dilation: $Dilation(I,\ K) = \cup \cup_{(x\prime,y\prime)\in K} I(x+x', y+y')$
  以 Kernel 中心為原點，對原圖進行位移(x',y')，最後對所有位移的圖取聯集或是最大值
- Erosion: $Erosion(I,\ K) = \cap \cap_{(x\prime,y\prime)\in K} I(x+x', y+y')$
  以 Kernel 中心為原點，對原圖進行位移(x',y')，最後對所有位移的圖取交集或是最小值
- Boundary Extraction: Erosion 的 kernel 選擇 $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$則會留下 Interior Pixel，即 8-connectivity 中的鄰域 pixel 都為 255。將原圖減掉 Interior pixel 後則會留下 Boundary

(b) Perform hole filling on sample1.png

result2.png

Reasoning: 因直接對圖片做 Hole Filling 不好定義初始點，於是先反轉圖片將背景填滿再反轉回來較容易

Approach: 對原圖做 Boundary Extraction 後把每個物件的區域都劃分開來，接著用 Hole Filling 把背景填滿，再對圖片取 Complement 反轉。反轉後背景與 Boundary 的區域會變為 0 所以需要把 Boundary 加回來。

Implementation:

- Hole Filling: $Fill(Image,\ Start\_Position)$

  Input: Image of size MxN, Start_Position(J,K)

  Output: Filled Image

  Procedure:

  1. G0 <- Copy of Image
  2. Set the starting point (J,K) equal to 255
  3. do until G0 == G1 or exceed max iteration
  4. $G1 \leftarrow Dilation(G0, \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix})$
  5. $G1 \leftarrow G1 \cap Image_c$
  6. $G0 \leftarrow G1$
  7. return G1

(c) Design an algorithm to remove background noise of sample1.png


result3.png

Reasoning: 對原圖做 Opening 即可去除此背景雜訊，透過 Erosion 將細小的白點移除後用 Dilation 將目標復原

Approach: 對原圖做 Opening(Kernel)，Kernel 為$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$的矩陣

Implementation:

- Opening(Image, Kernel): $Dilation(Erosion(Image, Kernel), Kernel)$

Discussion:

用 Median Filter 3x3 與 5x5 得到以下兩張圖。


Median 3x3


Median5x5

Median 3x3 的效果與 Opening 的結果類似，在小狗頭上的對話框中 Opening 可以保存背景的資訊而 Median filter 的部分的黑色會遺失。Median 5x5 會使較小的區域的資訊遺失。

(d) Design an algorithm to count the number of objects in sample1.png

Reasoning: 透過 Connected Component Labeling，可以將圖中相連的區塊標記，並計算區塊的總數。

Approach: Execute connected component labeling on source image and return the label map and label count

Implementation:

- Connected Component Labeling(Image)

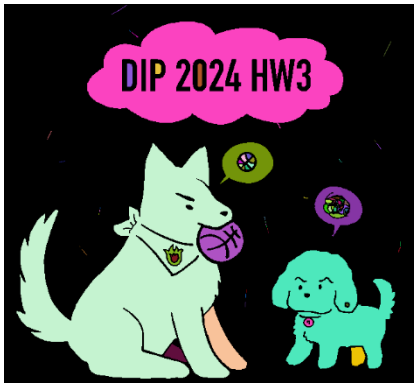
  Input: Image of size MxN

  Output: Label count and Label map

  Procedure:

  1. Let label = 0, Label Map = Negative one matrix of size MxN
  2. for starting point (p,q) where $(p, q) \in Image$ and Label Map[p,q] is equal to -1
  3. label <- label + 1
  4. G0 <- Zero array of size MxN
  5. Set the starting point (p,q) in G0 equal to 255
  6. do until G0 == G1 or exceed max iteration
  7. $G1 \leftarrow Dilation(G0, \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix})$
  8. $G1 \leftarrow G1 \cap Image$
  9. $G0 \leftarrow G1$
  10. Set the Label Map[j,k] = label where $\{ (j,k) \mid G1[j,j] = 255 \}$
  11. return label and label map

Result: 由於沒有去除雜訊，背景中的小白點亦為 Object，最後 label 數為 75




Label image

## Problem 2: TEXTURE ANALYSIS

sample2.png

(a) Apply Law's method to sample2.png to extract the feature vectors

Approach: First apply convolution with 9 filters to get 9 intermediate images M. Then compute features with

$$E_i(j,k) = \sum_{(j\prime,k\prime)\in W} M_i(j+j',k+k')\text{^}2$$

where $W$ is the window with size 13 x 13. The computation is also called Angular Second Moment.

Discussion:

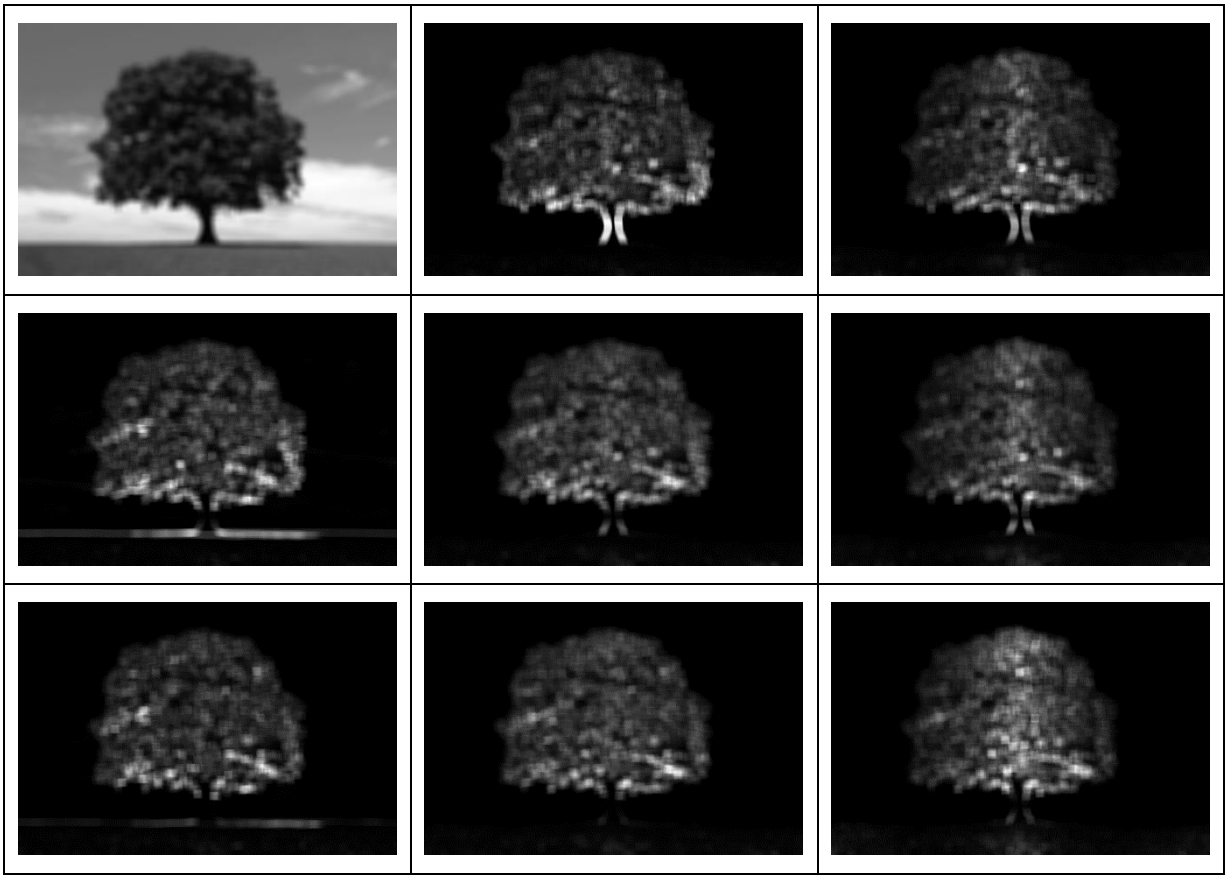


Table 1.

當 window size 為 13x13 時，可以將較多鄰近的 texture 包含在 feature map 中，

而當大於 13x13 時效益會下降且執行時間延長。Table 1 中左上到右下為經過 H1~H9 filter 與 Energy Computation 的圖片。因為需要將 Feature normalized to 255 才能顯示，較低值的部分會被壓縮到 0。第一張圖為相似於 Gaussian Filter 後的圖，為 low pass filter。3rd column 為垂直向的 high-pass filter，會抓取垂直向的特徵，而 3rd row 為水平向的 high-pass filter，會抓取水平向的特徵。

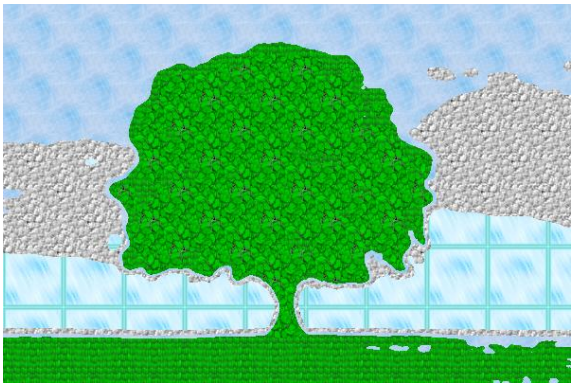
(b) Apply the k-means algorithm to classify each pixel using the feature vectors obtained in (a)

result4.png

Approach: Let 9 features from (a) be $T_1$ … $T_9$. Each pixel would be a vector $(t_1, \ldots, t_9)$. Apply k-mean to each pixel and find the k = 5 regions.

Discussion: 由原圖中可以用 supervised 的方式看出有大概五個 regions(天空、薄雲、厚雲、樹、草)，設 k = 5 去做 k-mean。Result4.png 中可以看到大部分的區域有被選到。其中樹上葉子部分中較亮會被判斷成地上的草，以及在地平線的附近的像素 feature 不好被辨認。

(c) Add different texture patterns to each region



result5.png

Approach: Texture 為 64x64 的 png 圖片，五個區域有各自的 texture。當 pixel 位於圖片中的(x,y)時對應到 texture 中(x % texture_height, y % texture_width)的位置