Digital Image Processing, Spring 2023

Assignment 3 - Report

R11922133, 廖政華, 資工所碩一

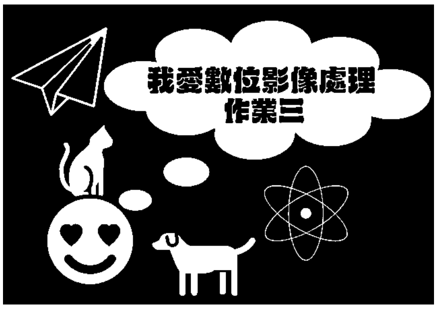
Problem 1: MORPHOLOGICAL PROCESSING

(a) Boundary extraction

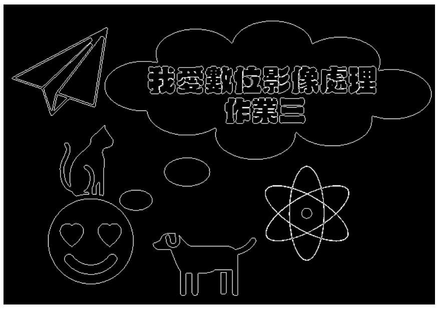
To extract boundary of elements in the given image I, one can first erode the image by a four-connected structuring element B, and then performing set differences between the original image and its erosion.

$$I - (I \ominus B)$$

.



(a) sample1.png



(b) result1.png

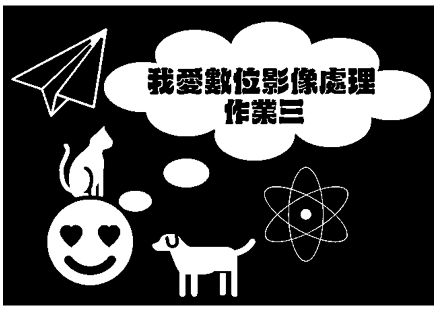
(b) Hole filling

I first obtain locations that are known to be holes by connected component labeling. To begin with, I form an array of zeroes X_0 and set the value of one of the locations belonging to each component to 1. Then, follow the iterative procedures

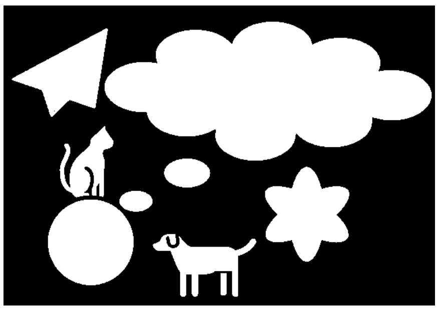
$$X_k = (X_{k-1} \oplus B) \cup I^C$$

, where B is a four-connected structuring element, until $X_k = X_{k-1}$. Finally, the hole-filled image is obtained by

$$I \cup X_k$$



(a) sample1.png



(b) result2.png

(c) Object counting

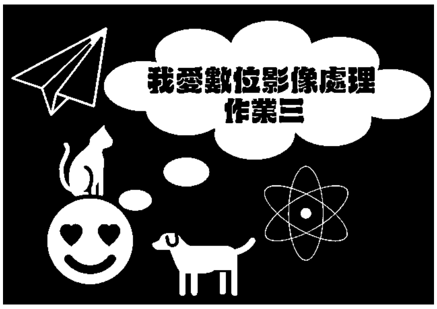
The object counting algorithm is described as the following iterative procedure. At termination, the iteration steps corresponds to the number of objects in the image.

1. Randomly pick a foreground pixel (j, k) from I as a starting point.
2. Initialize an array of zeroes X_0 and set the value of position (j, k) to 1. Then perform the connected component labeling procedure

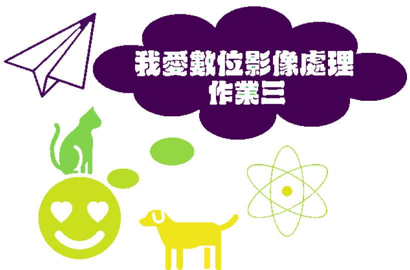
$$X_k = (X_{k-1} \oplus B) \cup I$$

   until $X_k = X_{k-1}$.
3. Perform set differences between I and X_k (i.e. remove the component found in this step from the image). If there are still foreground pixels present in the image, repeat this procedure from the first step; otherwise, terminate.

An object is defined by a connected component. By this definition, there are 38 objects in the given image.



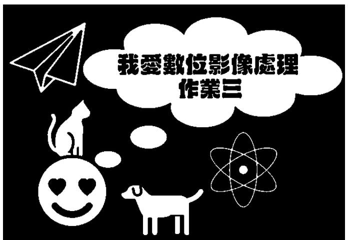
(a) sample1.png



(b) Labeled components.

(d) Open and close operators

While applying close operator appears to have minimal effect on the image, applying open operator with structuring element of larger connectivity results in removal of more small objects (such as the space within the Chinese characters) from the image.

	result3.png	result4.png
SE = four-connected		
SE = eight-connected		

Problem 2: TEXTURE ANALYSIS

(a) Laws method

For completness, I use three 1D masks that represent the low-pass, band-pass, and high-pass components

$$v_1 = \frac{1}{6}[\begin{array}{ccc} 1 & 2 & 1 \end{array}]$$
$$v_2 = \frac{1}{2}[\begin{array}{ccc} -1 & 0 & 1 \end{array}]$$
$$v_3 = \frac{1}{2}[\begin{array}{ccc} 1 & -2 & 1 \end{array}]$$

to compose the Laws' micro-structure impulse response arrays

$$G = \{v_i \otimes v_j | 1 \leq i, j \leq 3\}$$

.

The feature vector $F \in \mathbb{R}^{H \times W \times 9}$ is given by performing convolution on the input image I

$$F = I * G$$

.

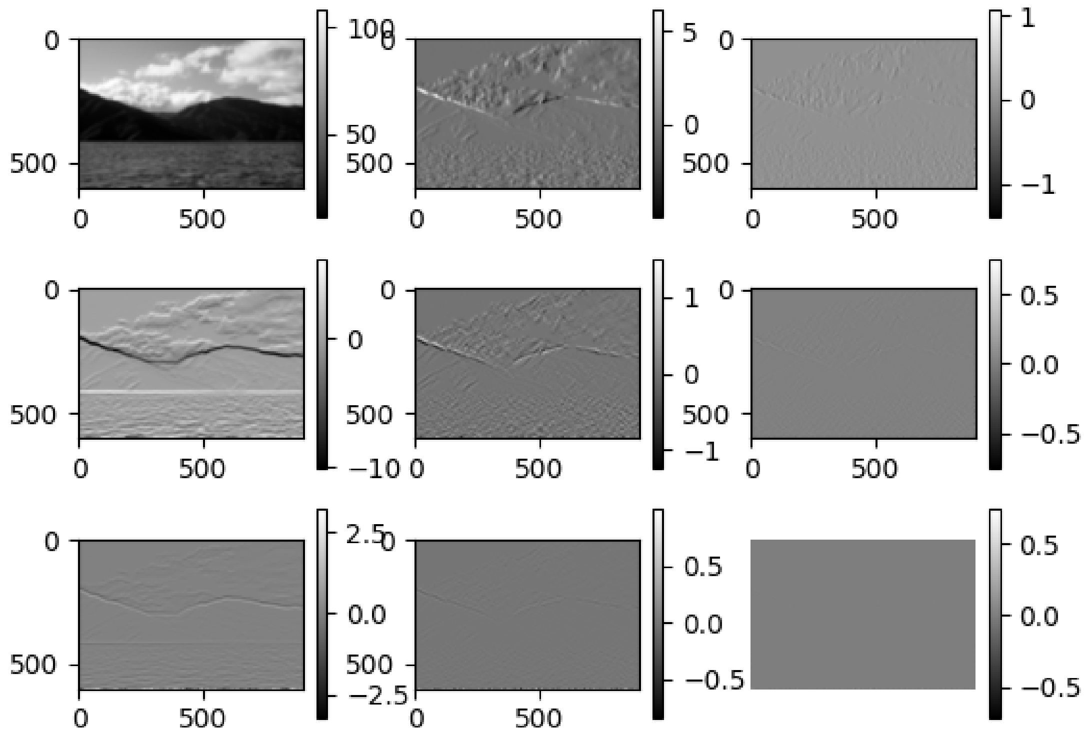
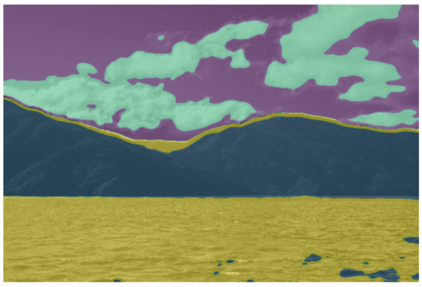


Figure 4: Visualization of feature vectors.

(b) Texture classification

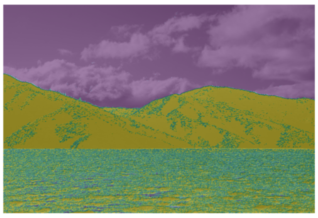
To classify the textures, we perform kmeans clustering on feature vectors obtain in Laws method with energy computation. In this case, the number of clusters is heuristically set as $k = 4$, in which it assumes that there are textures of the ocean, the mountain, the sky, and the cloud. Also, the kmeans++ algorithm is used for better initialization.

(a) sample2.png

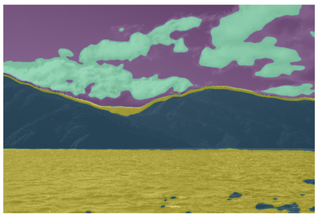


(b) result5.png ($r = 17$, `mean`)

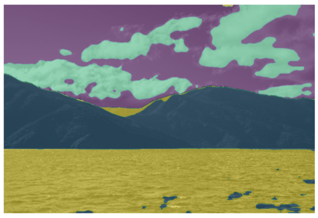
Effect of function used in energy computation. Results of using different function in the energy computation step are shown in Figure 6. It shows that `mean`, `median`, `correlation` and `inertia` functions produce better segmentations. For computation efficiency, the following experiments will consider the `mean` function.



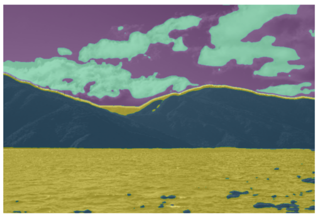
(a) no energy computation



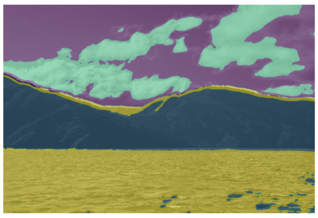
(b) mean



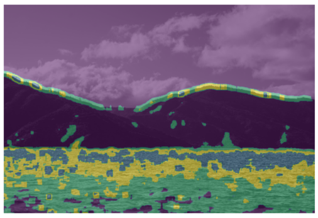
(c) median



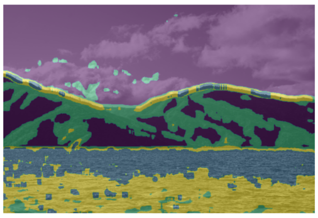
(d) correlation



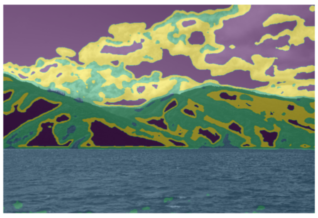
(e) inertia



(f) var



(g) std



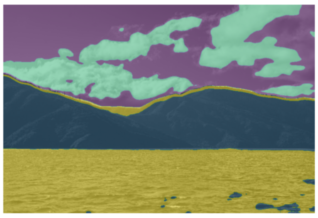
(h) entropy

(i) energy

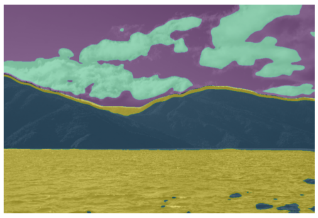
Figure 6: Result of different energy computation function ($r = 17$).

Effect of window size r in energy computation. Employing a larger window size generates coarser segmentation contours, whereas using a smaller window size may produce fragmented segmentations as it fails to capture local patterns owing to its limited receptive field. Thus, the window size should be carefully chosen according to the characteristics of different textures.

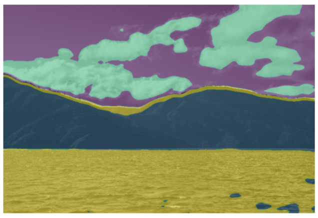
(a) $r = 3$



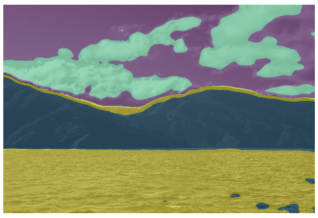
(b) $r = 17$



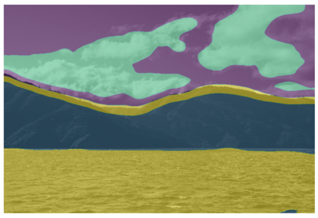
(c) $r = 19$



(d) $r = 21$



(e) $r = 23$



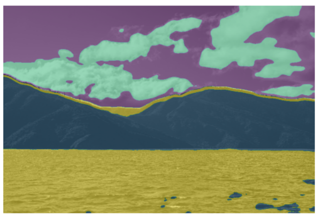
(f) $r = 51$

Figure 7: Result of different window size r in energy computation.

Feature importance. In fact, the same segmentation can be obtained using only the first feature (i.e. the Low-pass-Low-pass channel), as shown in Figure 9-(a). One possible reason is that the first feature has greater variance, causing the algorithm to align more along the first axis. However, using normalized features does not result in better segmentation, as shown in Figure 9-(b). This finding suggests that the first feature plays a more critical role in segmentation (in this particular setting). To illustrate this point, we can visualize the value distribution of each cluster, as depicted in Figure 10.

	0	1	2	3	4	5	6	7	8
count	540000.000000	540000.000000	540000.000000	540000.000000	540000.000000	540000.000000	540000.000000	540000.000000	540000.000000
mean	12.929021	0.001844	0.000014	-0.041549	-0.000055	-0.000019	0.000533	0.000029	0.000008
std	7.152667	0.245493	0.039484	0.646552	0.117167	0.029179	0.170427	0.070652	0.027238
min	2.622689	-2.106049	-0.503457	-4.389136	-0.963333	-0.543333	-1.338272	-0.692222	-0.575556
25%	6.599722	-0.092593	-0.013580	-0.200864	-0.038889	-0.008889	-0.027407	-0.012222	-0.006667
50%	10.685364	-0.009383	0.000123	0.007037	0.000000	0.000000	0.000247	0.000000	0.000000
75%	19.548021	0.076543	0.013951	0.166420	0.038889	0.008889	0.027901	0.012222	0.006667
max	27.136036	2.608765	0.411481	2.477778	0.984444	0.688889	1.338519	0.648889	0.585556

Figure 8: Statistics of each dimension of feature.



(a) segmentation using only the first feature

(b) segmentation with normalized feature

Figure 9

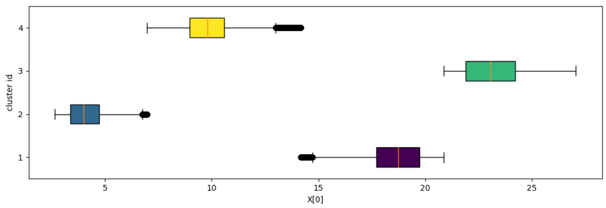


Figure 10: Distribution of values of the first feature for different clusters.

(c) Improving classification

There are mainly two things to improve in `result5.png`: First, pixels along the boundary of the mountain and sky are falsely classified as ocean. Second, some parts of the ocean in the lower right region are misclassified as mountain.

Considering the first problem, a possible explanation could be attributed to the phenomenon that only one feature is predominant in clustering. What happened is that the window surrounding the boundary pixels has a higher variance, as illustrated in Figure 11, and taking mean of these values happens to result in values that are similar to those of the cluster corresponding to the ocean texture (In fact, this is also indicated in Figure 10). Although using a smaller window size could address this problem, it does not produce good segmentation, as already pointed out in Figure 7.

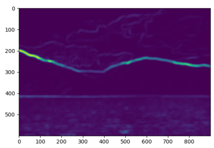


Figure 11: Variance of the first feature with window size $r = 17$.

One possible approach to tackle this issue is to introduce new features. It can be observed in `sample2.png` that the texture distribution seems to exhibits a certain spatial regularity. Specifically, each texture appears to be spread along the horizontal axis and is separated from others in the vertical axis. As a result, it seems reasonable to include the pixel coordinates in the feature vector. To achieve this, we introduce two new features:

$$F[j, k, 10] = w_j \frac{j}{H}$$
$$F[j, k, 11] = w_k \frac{k}{W}$$

The purpose of dividing by the height and width is for normalization. In terms of hyperparameters, we have two variables, w_j and w_k, which determine the relative "importance" of the horizontal or vertical coordinate in clustering. Following the above observations, we can select a larger value for w_j and a smaller value for w_k to reflect this pattern.

(a) sample2.png

(b) result6.png ($w_j = 0.7$, $w_k = 0.3$)

(d) Image quilting

Using the segmentation obtained in `result6.png`. It is easy to mark the ocean region and replace it with other texture. Specifically, it is replaced with synthesized texture derived from `sample3.png`. To carry out texture synthesis, I have implemented the image quilting algorithm as proposed in [1]. The resultant image, `result7.png`, was obtained with a patch size of 50×50, and the width of the overlap edge was set to $1/6$ of the patch size.

(a) sample2.png

(b) sample3.png

(c) result7.png (patch size = 50)

References

[1] Efros, A. A., and Freeman, W. T. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), pp. 341–346.