REPORT ON TEXTURE

Name: Trần Trường Giang
Student ID: BI9-090

I. Objective

The purpose of this lecture is to define the definition of texture, understanding its properties, and how to analyse texture.

II. Method

To analyse a texture image, first we need to find the co-occurrence matrix of the image. Open a grayscale image by open cv will return us an 8bit image(I) so the glcm(M) will have the size of 8x8.

$$M(i,j) = \sum_{x=1}^{n}\sum_{y=1}^{m}\begin{cases}1, & \text{if } I(x,y)=i \text{ and } I(x+\Delta x, y+\Delta y)=j\\ 0, & \text{otherwise}\end{cases}$$

Then we find the symmetric matrix by adding its transpose to itself. Then normalized by the sum of every element in the new symmetric matrix.

After we have the GLCM we can extract more features from the gray levels co-occurrence matrix.

Homogeneity: $\sum_{i,j=0}^{N-1=255} \frac{M_{i,j}}{1+(i-j)^2}$

Contrast: $\sum_{i,j=0}^{N-1=255} M_{(i,j)}(i-j)^2$

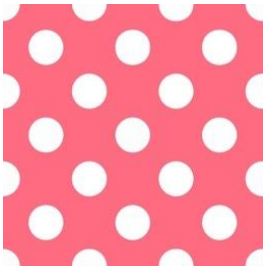
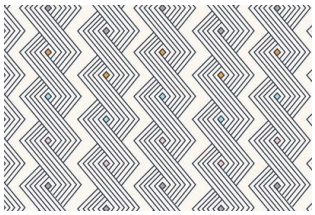
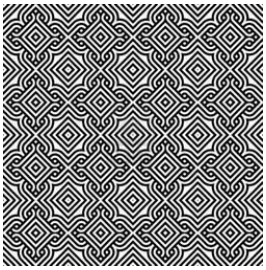
III. OpenCV Illustration

Code:

```
import cv2 as cv
import numpy as np
import math as m

img=cv.imread('test.jpg', cv.IMREAD_GRAYSCALE)


#Creating Grayscale level co-occurrence matrix
```

```
col = img.shape[0]
row = img.shape[1]
intglcm = np.zeros((256, 256),dtype=np.uint8)
glcm = np.zeros((256, 256),dtype=np.float32)
for i in range(0,col-1,1):
    for j in range(0,row-2,1):
        intglcm[img[i][j]][img[i][j+1]]+=1

intglcm = np.transpose(intglcm)+intglcm;
sum = 0
for i in range(0,255):
    for j in range(0,255):
        sum += intglcm[i][j]

for i in range(0,256):
    for j in range(0,256):
        glcm[i][j] = intglcm[i][j]/sum


#Some features extracted from the GLCM
energy = 0
homogeneity = 0
contrast = 0
entropy = 0
for i in range(0, 256):
    for j in range(0, 256):
        energy += np.float(glcm[i][j]*glcm[i][j])
        homogeneity += np.float(glcm[i][j]/(1+abs(i-j)))
        contrast += float((i-j)*(i-j))*glcm[i][j]
        if (glcm[i][j] != 0):
            entropy -= float(glcm[i][j]*m.log(glcm[i][j],10))


print("energy: "+str(energy))
print("homogeneity: "+str(homogeneity))
print("contrast: "+str(contrast))
print("entropy: "+str(entropy))
```

Test pattern:

Result:

	test.jpg	test2.jpg	test3.jpg
Energy	0.611853	0.005756	0.000526
Homogeneity	1.0708	0.2073	0.0595
Contrast	134.4	6777.5	11454.3
Entropy	1.16	3.79	4.73

IV. Conclude

The homogeneity is correlated with how repeatable the pattern is. The first contain only white dots, hence, the highest. Meanwhile the third pattern is complex and contains flowery shape which gives it the lowest homogeneity out of all.

For contrast the third pattern has the highest value because it black and white colors. In the opposite the fist pattern has the lowest value due to its colors is white and pink-ish.