EE475 **Fall 2019**
HW#5

1)Clustering: You will write a K-means clustering program, that takes an image as input, number of clusters (N) as argument and the segmented image as the output.

Follow the steps:

i) Randomly select N points from the image and consider their intensity values as the initial centers (means) $M(n)$ n=1,2,…N.

ii) Label each pixel in the image with the index of the center to which it is closer to:

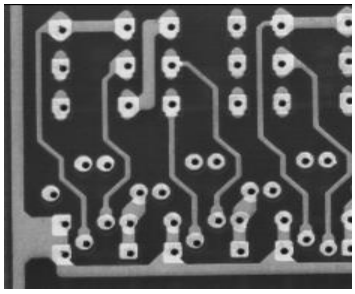
$$L(i,j) = \underset{n=1,2,..N}{\arg\min}\left(\left|I(i,j) - M(n)\right|\right)$$

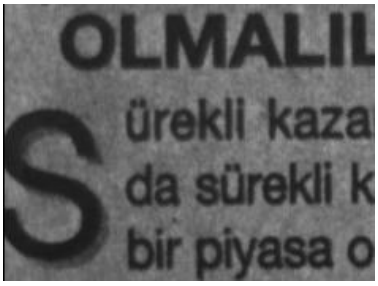
iii) Compute new means M(n) from the intensity values of the pixels labeled with n.

iv) Repeat (ii) and (iii) until M(n)'s tend to remain constant.

v) Obtain the image $M(i,j) = M(k) \quad where \quad k = \underset{n=1,2,..N}{\arg\min}\left(\left|I(i,j) - M(n)\right|\right)$, which is the segmented image.

Apply this program to "Gazete.bmp", "pcb.bmp" and starfish.bmp images and display the segmented images. You will decide on the number of clusters by observing the images and/or their histograms. For all images, try color only (i.e., gray level for black-and white images or RGB for color the image) features as well as color plus coordinate features, i.e, [Gray x y] and [R G B x y], respectively. Comment on the resulting segmentation maps.





2)Region growing: Use seeded region growing on the images: Gauss_rgb and Berkeley_horses. In other words, you plant region seeds appropriately. Label each seed with a different (positive) number (this will be the label of the corresponding regions). For the "horses" image, use the three-segment version User #1105
https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/BSDS300/html/dataset/images/color/113016.html

a) Centroidal growing: Let the regions grow according to the color predicate. For each unlabeled pixel p, use Euclidean distance from the test pixel to the region centroid. The distance must be compared with the threshold Th. The centroid is simply the average of the region pixels, that is, $\boldsymbol{c} = \frac{1}{|Reg|}\sum_{(i,j)\epsilon Reg}\begin{bmatrix} R(i,j) \\ G(i,j) \\ B(i,j) \end{bmatrix}$, where $|Reg|$ is the cardinal of Reg, the number of pixels in a region at that stage of growth and the predicate is: $\sqrt{(R(p)-c_R)^2+(G(p)-c_G)^2+(B(p)-c_B)^2} < Th$. If the pixel does not satisfy the predicate, then leave p unlabeled.

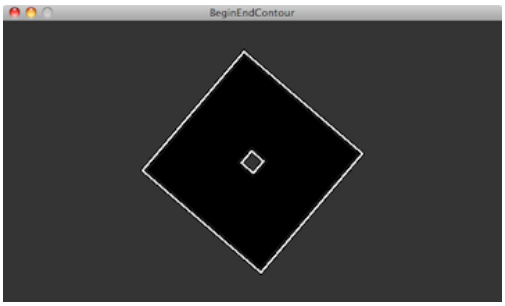
b) Recursively repeat b) until no change occurs in the labels (i.e. regions do not grow further).

c) Use the threshold Th = Th + Δt, repeat b-c until no pixels are left unlabeled. You have to decide for the threshold: For example, you may start with with Th = 5 and Δt = 10

d) Finally, compare the resulting segmentation map with the ground truth map. One goodness criterion could be (G: ground-truth region set, S: segmented region set) $IoU = \frac{|G \cap S|}{|G \cup S|}$, where IoU means Intersection over Union, one for each region.

3)Edge Detection

Generate a 256x256-sized image composed of two concentric equilateral diamonds. The inner diamond has edges 64 pixels long, mean gray value 192 and is centered at (128,128 or 127x127 if you start counting from (0,0)); the outer diamond should have edges 128 pixels long and mean gray value of 128 (i.e., in the inter-diamond band). The background has mean value 64. Consider the noiseless image (D_o) and then the noisy image contaminated with zero-mean, variance 144 Gaussian noise (D_g). In either case, you know the ground-truth edges.



(Note: This image is just for illustration. You must generate images yourself)

For edge detector performance, EP, use the measure given below; where the delta-Dirac $\delta(e_{grt}, e_{exp})$ means that there is an edge found on the ground-truth location; and $I(0 < edge\ distance \leq 2)$ predicate signifies that the experimentally determined edge is offset by at most two pixels from the corresponding ground-truth edge location. N_{grt} is the number of ground-truth edge pixels:

$$\text{EP}= \frac{1}{N_{grt}} \sum_{i=1}^{N_{grt}} \delta(e_{grt}, e_{exp}) + 0.5(1 - \delta(e_{grt}, e_{exp}))I(edge\ distance \leq 2)$$

$$\delta(e_{grt}, e_{exp}) = \begin{cases} 1 & if\ e_{exp}\ is\ colocated\ with\ e_{grt} \\ 0 & if\ no\ edge\ at\ the\ e_{grt}\ location \end{cases}$$

a) Compare the performances of the (a) Sobel, (b) Laplacian of Gaussian (LoG), (c) Canny edge detectors, by plotting the edge maps. Choose black for edge, white for non-edge. For the LoG use the algorithm discussed in the class, i.e., find the LoG filter output, extract the zero crossings, eliminate zero-crossings with weak gradient magnitude. For Sobel and Canny, use the MATLAB version.

b) Add Gaussian noise *N(0, 484)* to the Diamonds image and compare the performance of the edge detectors by tabulating the EP scores.

4)Hough Transform

Use the edge field of the Diamond images found above, using the Canny edge detector. Then implement the Hough algorithm to find all the lines in the images. Take the extent of theta as $(-\frac{\pi}{2}, \frac{\pi}{2})$ and the distance, from the origin in the (0, 512√2) range for the line equation $\rho = x\cos\theta + y\sin\theta$. Decide for the quanta sizes of the accumulator; for example, you may set the angular resolution at 5 degrees and distance from the origin in steps of 8 pixels, though this is only a suggestion.

Find the peaks in the Hough transform, and draw “red” lines over the edge map with line parameters corresponding to the Hough peaks.