Melik Burak Bozbey
2015401117

EE475 – HW5

1) For "Gazete.bmp", by using its histogram and observing the image, k = 2 was selected for segmentation with respect to gray level intensities for preserving characters and background:

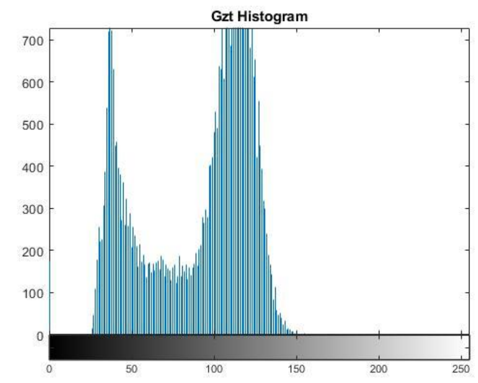


Fig 1. - "Gazete.bmp" Histogram

For "pcb.bmp", by using its histogram and observing the image, k = 3 was selected for segmentation with respect to gray level intensities to preserve pinholes, lines and background:

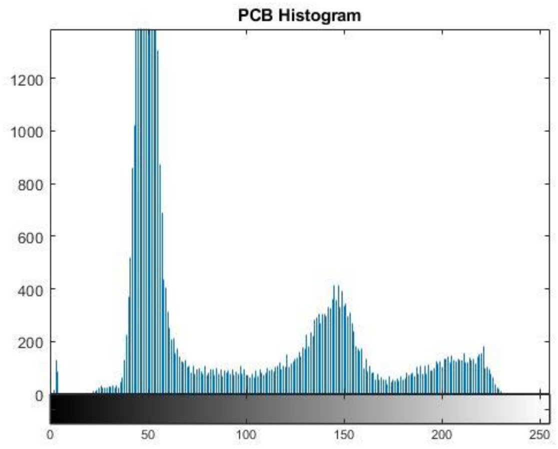


Fig 2. - "pcb.bmp" Histogram

For "starfish.bmp", by observing the image and its colors, k = 6 was selected for segmentation with respect to RGB intensities:

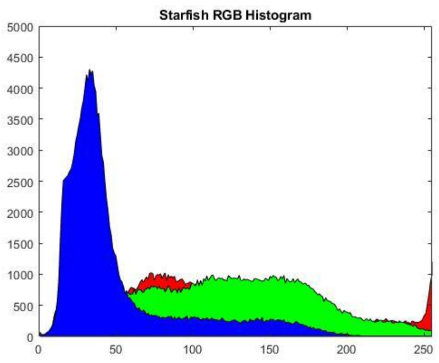


Fig 3. - "starfish.bmp" Histogram

After implementation of the algorithm, results of clustering with color only features are as follows:

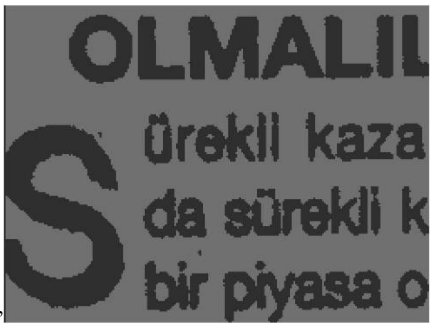
”

Fig 4. - “Gazete.bmp” - Color Only Features

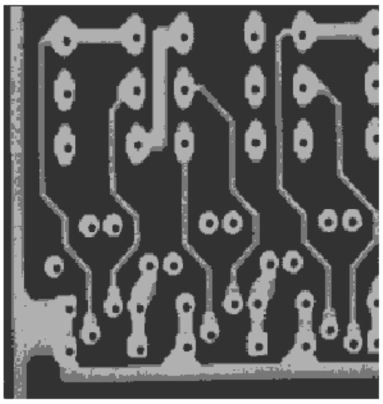


Fig 5. - “pcb.bmp” - Color Only Features

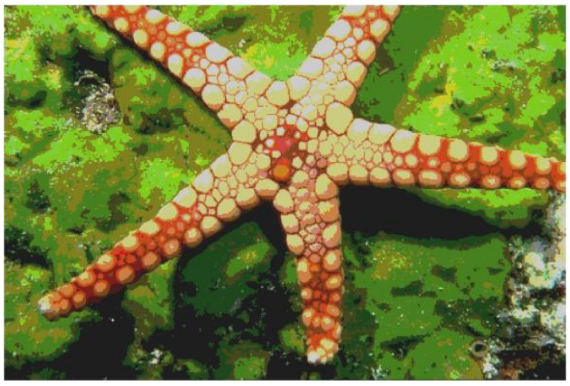


Fig 6. - “starfish.bmp” - Color Only Features

After implementation of the algorithm with both color and coordinate features and $k = 32$ centers for each image, the results are as follows:

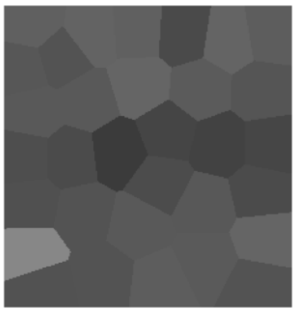


Fig 7. - “pcb.bmp” - Color + Coordinate Features

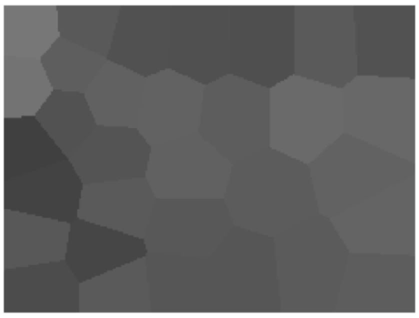


Fig 8. - "Gazete.bmp" - Color + Coordinate Features

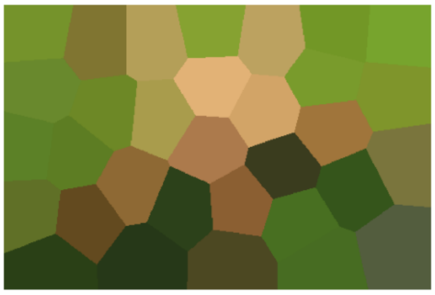


Fig 9. - "starfish.bmp" - Color + Coordinate Features

Since color only clustering gives clean segmentation of lines in "pcb.bmp" and letters in "Gazete.bmp", segmentation map was more representative with respect to these objects, rather than clustering result with coordinate features. It's because the objective should be identification of color specific parts such as lines & pinholes or characters. For "starfish.bmp", since there are different number of peaks in R,G and B histograms and different colors exist in the image, segmentation maps for this image are different than the others for color only features. When we add coordinate features to color features, given algorithm creates segments with respect to local distance from centers and colors. This method can be useful when we want to find out top-k dominant gray level for black and white images or RGB for color images.

2) After part a, b and c implemented, 3 seed were used for Gaussian_rgb image. To achieve maximum converge, another criteria was added. If there exists unlabeled pixels and threshold increased and neighborhood of a region centroid was labeled, neighborhood size increased from $n^2 - 1$ to $(n+1)^2 - 1$ $\quad for\ i = 3,5,7 \ldots 2k+1$ until an unlabeled pixel in neighborhood is found. Result of region growing in Gaussian_rgb image and it's edges are as follows:

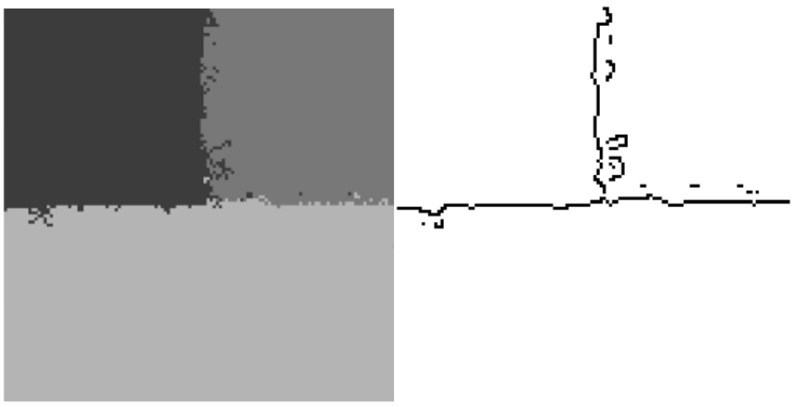


Fig. 10 – Segmented Image after Converge & Edge Detection with Sobel Detector

For Berkeley_horses image, only 2 seed was used for each horses due to computational burden. After 2 regions on horses are segmented until no improvements in regions, unlabeled pixels were marked as background. So, 3 regions were segmented with 3 seeds. Result of region growing in Berkeley_horses image and it's edges are as follows:

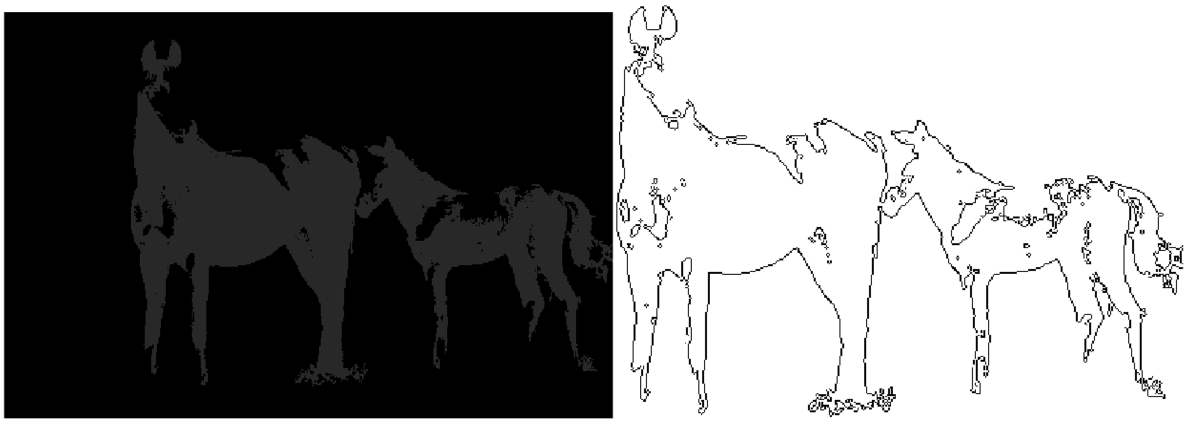


Fig. 11- Segmented Image & Edge Detection with Sobel Detector

IoU scores with respect to ground truth segmentation maps are as follows:

Area	Horse Left	Horse Right	Background
Segmentation Map 1	0.6543	0.4253	0.8682
Segmentation Map 2	0.6570	0.3730	0.3598
Segmentation Map 3	0.6526	0.4219	0.8676
Segmentation Map 4	0.6754	0	0.4902
Segmentation Map 5	0.7014	0.3609	0.4881
Segmentation Map 6	0.6690	0	0.4875

Table 1 – IoU Scores w.r.t Ground Truth Maps

- **Note**: Since there are multiple regions in some ground truth segmentation maps, 0 IoU scores exist in table which stem from region ambiguity in given segmentation maps because I only have 3 regions for background and two horses. I tried to merge ground truth segmentation maps with more than 3 regions to 3 regions w.r.t my segmentation map but for Segmentation Map 4 & 6, regions for horse in the right were not merged successfully.

3.a) After diamond image in Figure 12 was generated with given specifications, edge detectors were implemented to find edges on noisy image. The output of edge detectors are plotted in Figure 13.

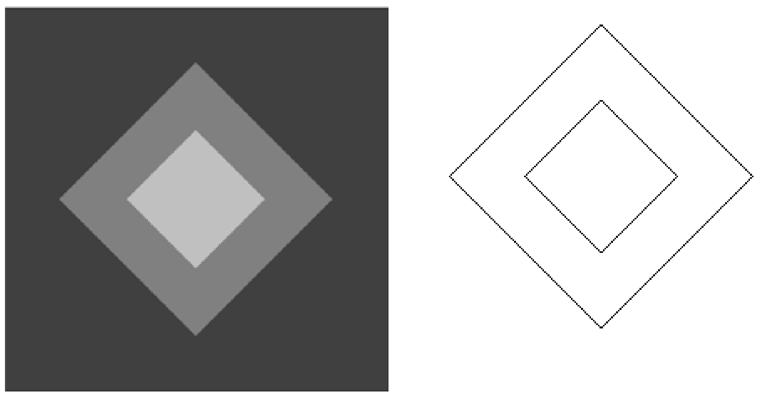


Fig 12. – Diamond Image & Ground Truth Boundaries of Diamond Image

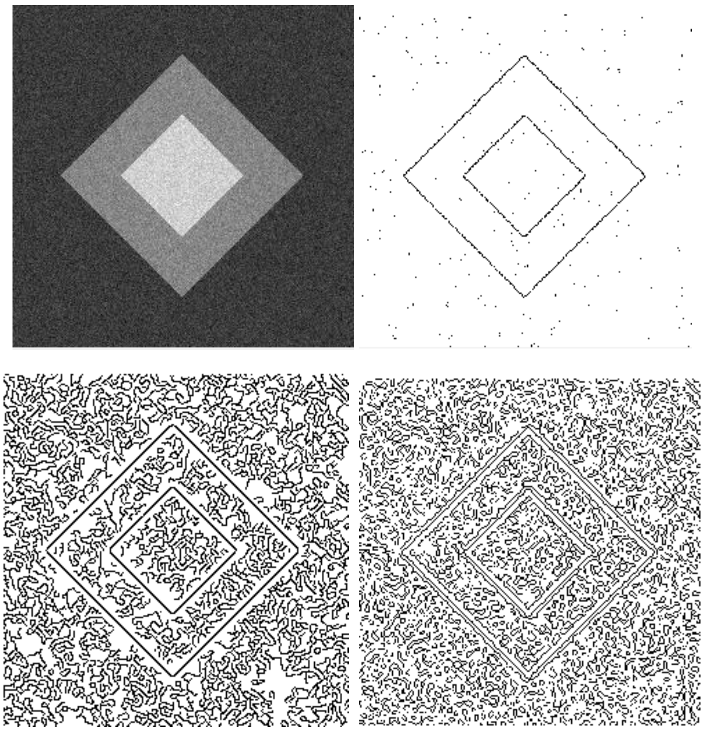


Fig 13. – (top) Zero Mean, 144 Variance Gaussian Noise on Diamond Image & Sobel Edge Detector on Diamond Image

(bottom) Canny Edge Detector on Diamond Image & LoG Edge Detector on Diamond Image

Performances of the given edge detector with respect to EP metric are as follows:

Edge Detector	**Sobel**	**Canny**	**LoG**
EP	440	983	1272

- From table, we can say that for diamond image with noise variance 144, LoG is better than Canny and Canny is better than Sobel.

3.b) Same procedure was applied as in part 3.a. Results are as follows:

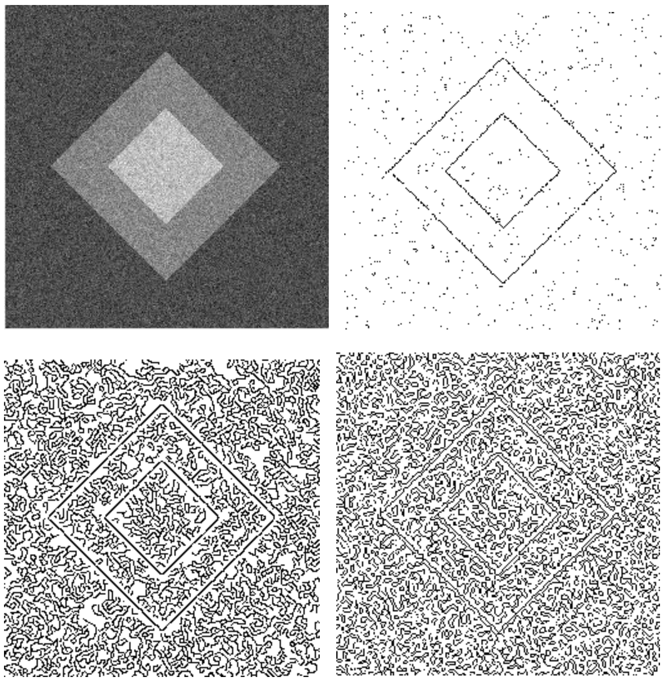


Fig 14. – (top) Zero Mean, 484 Variance Gaussian Noise on Diamond Image & Sobel Edge Detector on Diamond Image

(bottom) Canny Edge Detector on Diamond Image & LoG Edge Detector on Diamond Image

Performances of the given edge detector with respect to EP metric are as follows:

Edge Detector	**Sobel**	**Canny**	**LoG**
EP	407	528	322.5

- From table, we can say that for diamond image with noise variance 484, Canny is better than Sobel and Sobel is better than LoG. Since given metric also includes undetected but closely located edges with respect to ground truth, the performances for two noisy image were different. Since Canny detector gave more robust results on high variance, It's the best in this case.

4) Angular resolution was selected as 1 degree and minimum length for Hough lines was selected as 40 pixel. After implementation, the detected edges are plotted on the output of Canny Edge Detector on noisy image. Four edges were detected successfully and plotted with green lines. Results are as follows:

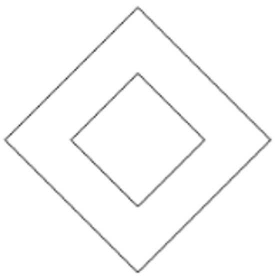


Fig 15. **–** Edge Image

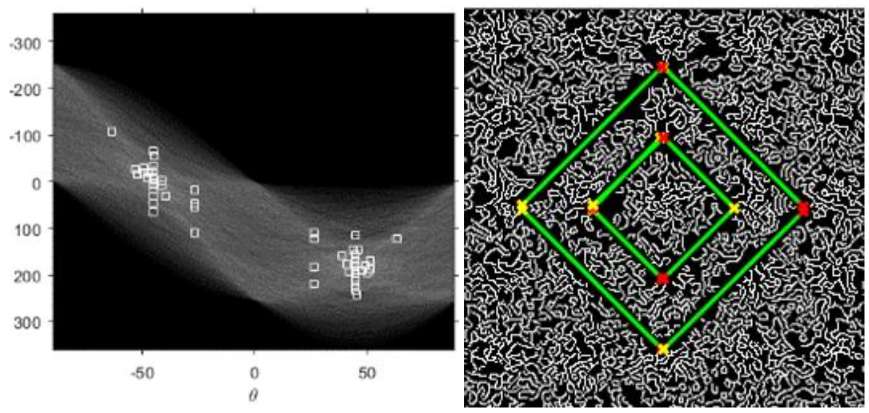


Fig 16. **–** Hough Peaks on Graph & Detected Lines on Diamond Image