Sefa Kayraklık
2015401066

# Homework 5

***Abstract*—This document contains the solutions, explanations, comments, and related graphs and tables of the questions in the homework 5.**

## I. Clustering

Clustering can be used to segment an image in terms of its color components or the coordinate features of the pixels with addition of color components. 3 images to examine their segments are shown in the Figure 1.

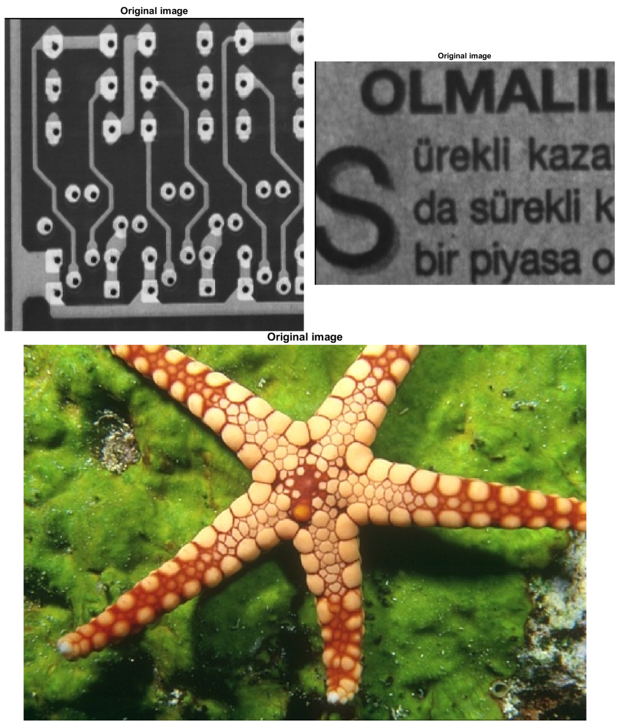


Fig. 1. The original images of PCB, gazete, and starfish, respectively

Clustering is implemented by the following procedure. First, $N$ points are selected randomly to be as initial centers (means), $M(n)$, $n = 1, 2, ..., N$. Then, each pixel in the image is labelled with the index of the center to which it is closer to. After labelling, the means $M(n)$'s are updated with the mean values of the pixels labeled with n. This procedure is repeated until the means remain constant. Finally, the segmented image, $M(i, j) = M(k)$, where $k$ is the index of the centers, are obtained.

As mentioned above, the feature vector can be only the color values of the pixels or their coordinate features with addition of color values.

**Feature vector with only color:** Only the color values are taken into account when the means of the centers are computed and initialized. So, the images are segmented according to their color values. Therefore, with this method, images can be separated as background and foreground. The result images with segments and their means of the given images are shown in the Figure 2 and 3.

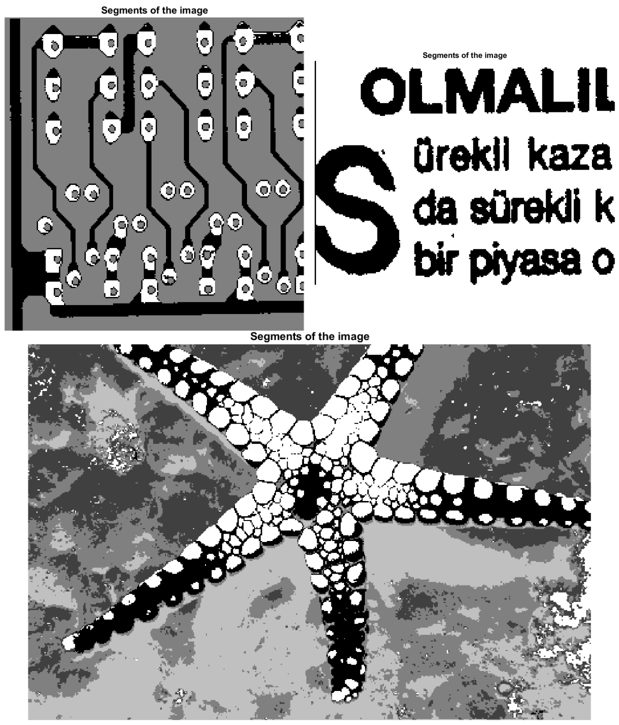


Fig. 2. The segmented images of PCB, gazete, and starfish, respectively

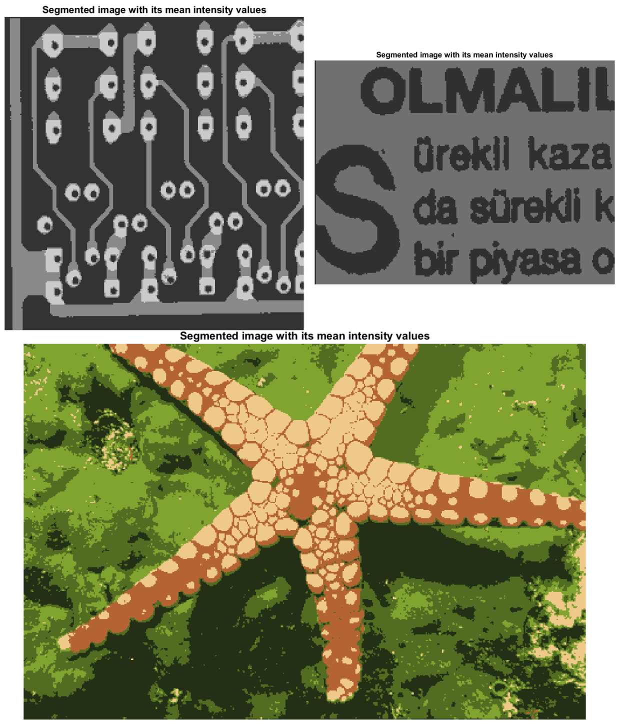

Segmented image with its mean intensity values
Segmented image with its mean intensity values
Segmented image with its mean intensity values

Fig. 3. The segment means of PCB, gazete, and starfish, respectively

In the PCB image, $N$ is selected as 3 since there exist 3 peaks in its histogram. The segments correspond to conducting paths, legs of the components and background. Actually, there should be another segment for the black dots on the legs; but, since their gray levels are closer to the background's, they are segmented as background. So, the PCB image is segmented very well with the color values of the pixels.
In the gazete image, $N$ is taken as 2 due to the same reasoning mentioned above. The segments correspond to the background and the letters. So, the writings given on a paper can be extracted from the background by using clustering method.
In the starfish image, $N$ is taken as 5 since the image is mostly formed by 5 distinct colors. So, the segments 1, black, and 5, white, correspond to the foreground, namely the starfish; the remaining segments correspond to the background, namely the mosses. So, the clustering method are also able to segment the color images.

**Feature vector with color and coordinates:** The color values and the coordinate information of the pixels are considered together in order to determine the feature vector of the clustering. Since the color and coordinate do not contribute to the segments with the same amount, they should have different weights. However, in order not to complicate the algorithm, their values can be just normalized. With the coordinate information, the image can be separated into different regions. The result images with segments and their means of the given images are shown in Figure 4 and 5.

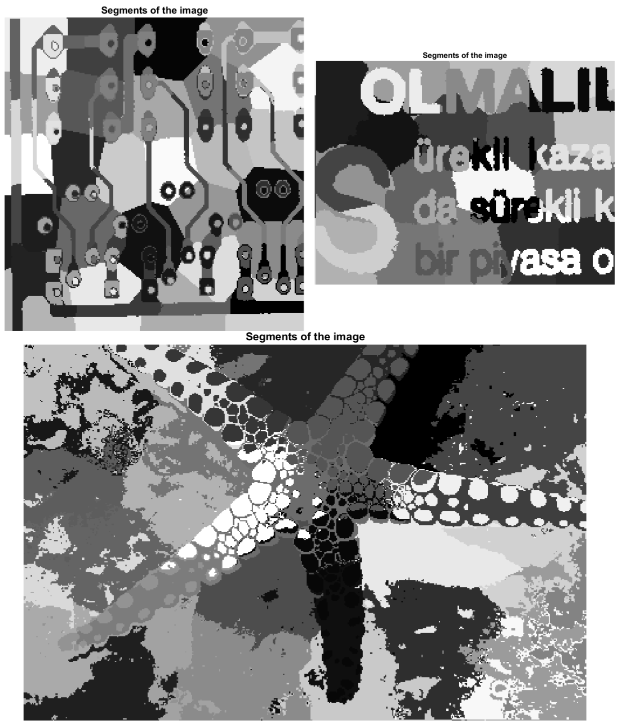
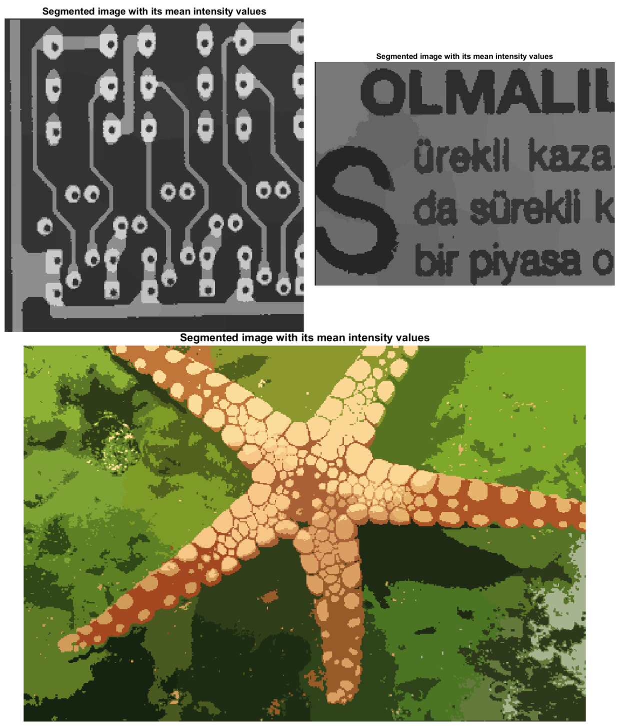


Fig. 4. The segments images of PCB, gazete, and starfish, respectively


Segmented image with its mean intensity values
Segmented image with its mean intensity values
Segmented image with its mean intensity values
Fig. 5. The segment means of PCB, gazete, and starfish, respectively

In the PCB image, $N$ is selected as 45 in order to obtain legs and conducting paths of the different region of the image. With this implementation, the legs in the different regions can be identified; for example, the legs on the right-bottom, left-bottom, right-top and left-top are in the different segments. As similarly, the conducting paths can be distinguished from each other. And, the background can be constructed by combining the corresponding background segments.
In the gazete image, $N$ is taken as 30 in order to obtain the the letters and the words of the different region of the image. With this implementation, the letter can be distinguished; for example, the *S* can be extracted by combining its 2 segments, *OLMALIL* can be separated as the letters *OL*, *MA*, and *LIL*.
In the starfish image, $N$ is taken as 34 to obtain different regions of the starfish. So, the different arms, and its circles of the starfish can be separated from each other; for example, the arm on the right-bottom is in the different segment from other arms. If the corresponding segments are combined, each arm of the starfish can be obtained, separately. And, with this implementation, the background is demonstrated more clearly in comparison to the only color segmentation. So, If the number of clusters and the weights between color values and coordinate information are determined appropriately, each desired segments on the image can be distinguished.

## II. Region Growing

Region growing can be used to separate an image into its different regions according to their color values. 2 example images to examine their regions are shown in the Figure 6.

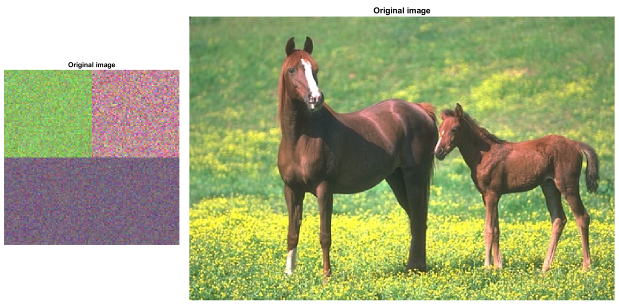


Fig. 6. The original images of gauss rgb and horses

Region growing is implemented by the following procedure. First, the seeds for the region are determined wisely. Then, the regions are expanded from its seeds according to the Euclidean distance from the color value of the unlabelled pixel to the color value of the region centroid, the mean of the region pixels. In this comparison, there exists a threshold value that decides whether the pixel belongs to the region or not by simply comparing the distances. If all the pixels which can be labelled by the initial threshold, then, the threshold is increased by some amount that should be determined. So, this procedure is repeated until no pixels are left unlabelled.
The resulting images with regions and their means of the given images are shown in the Figure 7, 8.

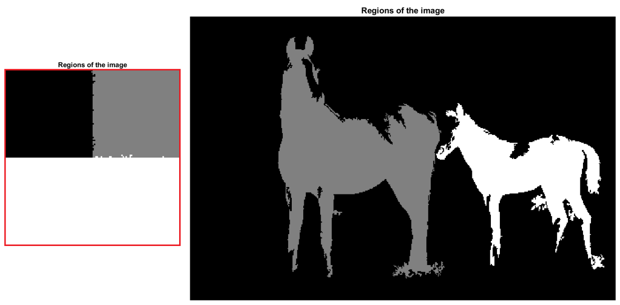


Fig. 7. The regions of images of gauss rgb and horses

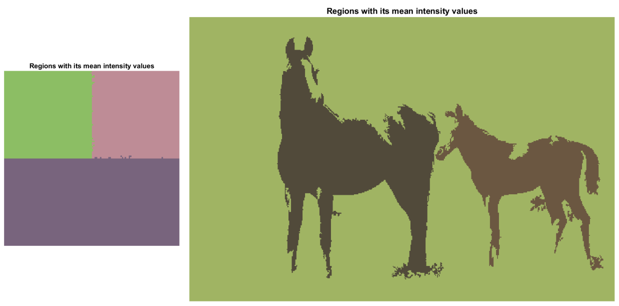

Regions with its mean intensity values
Regions with its mean intensity values

Fig. 8. The region means of gauss rgb and horses

The ground truth images of the given images are shown in the Figure 9.

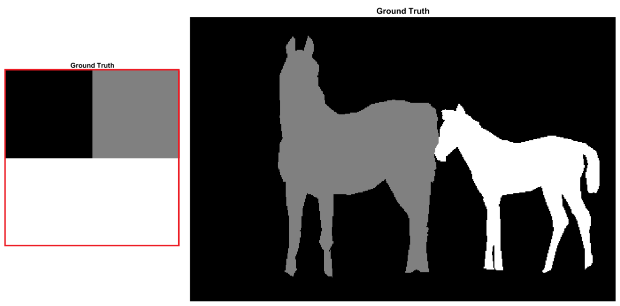


Fig. 9. The ground truth of gauss rgb and horses

In the gauss rgb image, the seeds are selected as $(32, 32), (32, 96), (96, 64)$ for the green, pink, and purple regions, where the first entry is the row, and the second one is the column of the pixel and the origin is the left-top. And, the initial threshold is determined as $5$ and the update of the threshold is $10$ increment in each loop. In order to determine the performance of the algorithm, the Intersection over Union (IoU) can be used. IoU is computed as the division of the intersection of the ground truth image and the resulting image, with the union of them. Namely,

$$IoU = \frac{|G \cap S|}{|G \cup S|}$$

With this metric, the performance of the algorithm in the gauss rgb image is $0.9959$ $0.9929$ $0.9985$ for the black region, the gray region, and the white region, respectively. So, it can be seen from the IoU values that the algorithm is very successful in labelling the gauss rgb image.

In the horses image, the seeds are selected as $(34, 326), (144, 144), (170, 382)$ for the background, big horse, and the small horse. And, the initial threshold is determined as $5$ and the update of the threshold is $19$ increment for the background and the big horse in each loop. Since the transition of the regions is not the same for each boundary, multi-threshold should be considered. So, the threshold increment for the small horse is $15$. With IoU metric, the performance of the algorithm in the horse image is $0.9417$ $0.8426$ $0.7836$ for the black region, the gray region, and the white region, respectively.
The performance of the algorithm can be increased by selecting more promising seeds for the region and by tuning the threshold values within some ranges. The parameters used above are selected heuristically, not by parameter tuning.

## III. Edge Detection

The Laplacian of Gaussian (LoG), Sobel, and Canny methods can be used to detect edges in an image. 2 example images to examine their edges are shown in the Figure 10. The one of them is the version of the contaminated with the zero mean, 484 variance Gaussian distribution.

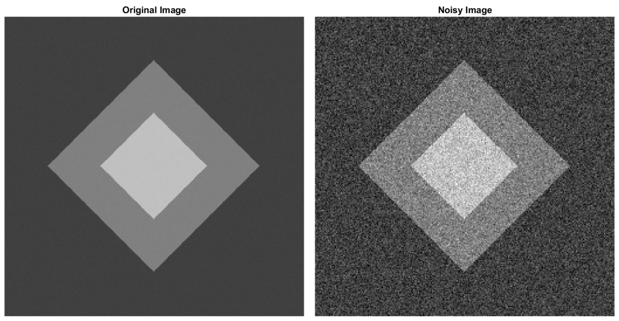


Fig. 10. The original image and the noisy image

LoG is implemented by the following procedure. First, the image is filtered with an $13*13$ LoG window with $\sigma = 2$. Then, the zero crossings of its output are selected as candidates of the edges. And, the ones with weak gradient magnitude, namely the ones which have gradient magnitude less than 30, are eliminated to obtain the actual edges. For the calculation of the gradient, an $3*3$ gradient window is used.
For Sobel and Canny, the MATLAB versions are used to detect the edges of the given images.

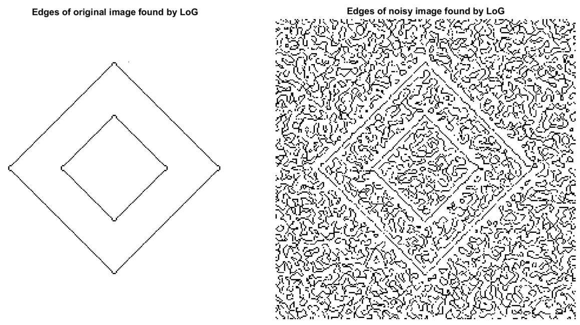

Edges of original image found by LoG
Edges of noisy image found by LoG

Fig. 11. The edges found the LoG algorithm

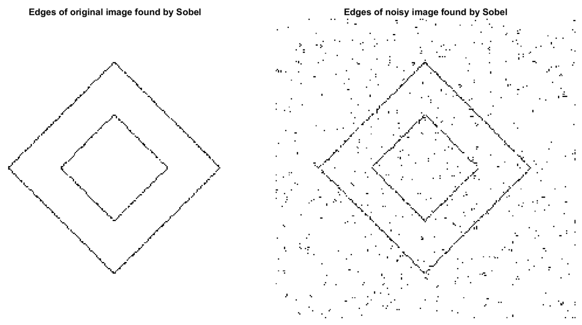

Edges of original image found by Sobel
Edges of noisy image found by Sobel

Fig. 12. The edges found the Sobel algorithm

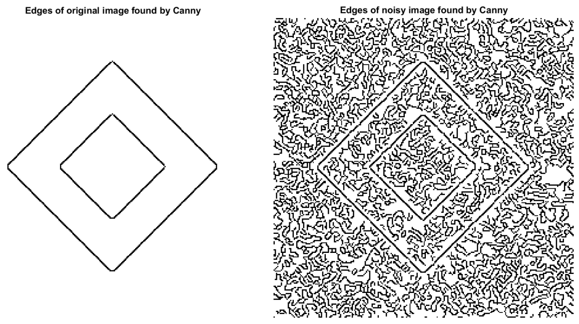

Edges of original image found by Canny
Edges of noisy image found by Canny

Fig. 13. The edges found the Canny algorithm

The Figure 11, 12, and 13 show the detected edges of the given images by using LoG, Sobel, and Canny, respectively.

In order to compare the performance of the algorithms with each other, the following edge detector performance metric, $EP$ is used.

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

,where $\delta(e_{grt}, e_{exp})$ means that there is an edge found on the ground truth location, $I(ED \leq 2)$ predicate signifies that the experimentally determined edge is offset by at most 2 pixels from the corresponding ground truth edge location, $N_{grt}$ is the number of ground-truth edge pixels.

Therefore, the following table shows the $EP$ values of the algorithms that are used in the original and the noisy image.

| Method | Original Image | Noisy Image |
|---|---|---|
| LoG | 0.9852 | 0.8639 |
| Sobel | 0.8102 | 0.7824 |
| Canny | 0.9981 | 0.9843 |

So, it can be seen from the table that Canny method gives the best EP whereas Sobel gives the worst EP. Since the interest is only the pixels which are labelled as edges in the ground truth, Canny and LoG are more successful than Sobel in both original and noisy image. However, if all the pixels are taken into account, the performance of the Sobel in the noisy image will be the best among the three, since it can distinguish the edges and the noise more effectively than the two other methods.

## IV. Hough Transform

Hough transform can be used to identify the lines on an edge map image by a voting procedure. 2 example images which are given in the Figure 10. In order to detect the edges and to form the edge maps, Canny method is used. So, the corresponding edge maps are given in the Figure 13.

In identification of the lines from an edge map, the following procedure is implemented. First, the edge map is transformed by Hough transformation. It draws the curves which are defined by the equation

$$\rho = x_i cos(\theta) + y_i sin(\theta)$$

for every pixel, where $i$ is the pixel indices and $\rho$, $\theta$ are quantized (in this case, they are quantized with 0.5). Then, the algorithm votes the corresponding quantized grid by one if it contains a part of a curve. After obtaining the image which is generated by voting, the algorithm finds the peaks, the most voted grids. So, from the corresponding $\rho$ and $\theta$ values of the pixels, the lines are drawn by using the following equation

$$y = -\frac{cos(\theta_j)}{sin(\theta_j)}x + \frac{\rho_j}{sin(\theta_j)}$$

, where $j$ is the peaks indices.

The Hough transforms and the peaks of the given edge maps are shown in the Figure 14. The negative of the shown images are plotted not to waste so much toner.

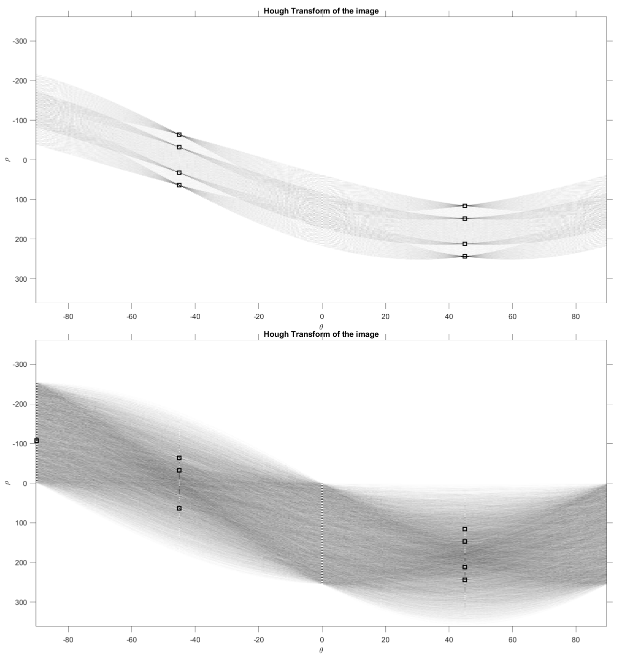


Fig. 14. The Hough transforms and the peaks of the edge maps of the original and noisy image, respectively

By using the peaks found in the Hough transformed edge maps of the given images, the lines are drawn and shown in the Figure 15. The red lines are the found Hough lines, the blue and green crosses are the starting and the ending of the lines, respectively.

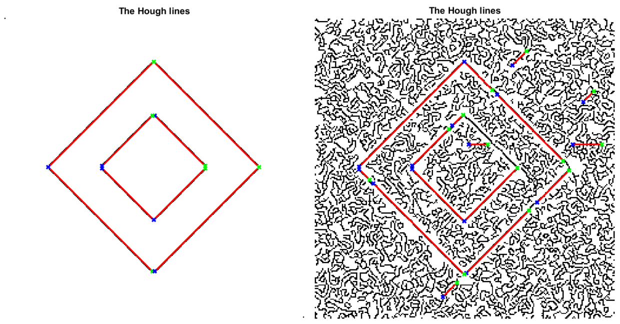


Fig. 15. The Hough lines of the edge maps of the original and noisy image, respectively

So, it can be seen from the figure that the Hough algorithm can find all the 8 lines in the edge map of the original image, since the 8 peaks of the Hough transform are found correctly; whereas it can find all the lines, except one, in the edge map of the noisy image, since one peak of the Hough transform is found incorrectly due to the existence of noise. And this wrong peak corresponds to the horizontal lines in the image.