CSE 573 - Project 3

Atrayee Nag (#50288651)

December 2018

1 Morphology image processing

a) Using two morphology image processing algorithms to remove noises of the image below, save the results and name as 'res_noise1.jpg' and 'res_noise2.jpg'. (2')

Morphological Image Processing:
Morphological image processing constitutes non-linear operations on images, which relates to the shape or morphology of the features in an image. Morphological techniques probe an image with a small shape or template called a **structuring element**. The structuring element is positioned at all possible locations in the image and it is compared with the corresponding neighbourhood of pixels. Our algorithm to remove noise from the images, constitute the two fundamental morphological operations:

1. Dilation:
The dilation of an image f by a structuring element s (denoted f $\oplus$ s) produces a new binary image g = f $\oplus$ s with ones in all locations (x,y) of a structuring element's origin at which that structuring element s hits the the input image f, i.e. g(x,y) = 1 if s hits f and 0 otherwise, repeating for all pixel coordinates (x,y).

2. Erosion:
The erosion of a binary image f by a structuring element s (denoted f $\ominus$ s) produces a new binary image g = f $\ominus$ s with ones in all locations (x,y) of a structuring element's origin at which that structuring element s fits the input image f, i.e. g(x,y) = 1 is s fits f and 0 otherwise, repeating for all pixel coordinates (x,y).

Compound Operations:
Many morphological operations are represented as combinations of erosion, dilation, and simple set-theoretic operations such as the complement of a binary image:

1. Opening:
The opening of an image f by a structuring element s (denoted by f $\circ$ s) is an erosion followed by a dilation: f s = (f $\ominus$ s) $\oplus$ s.

2. Closing:
The closing of an image f by a structuring element s (denoted by f $\bullet$ s) is a dilation followed by an erosion: f $\bullet$ s = (f $\oplus$ s) $\ominus$ s.

The **structure** element we have used to perform the compound operation is:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

For image **res_noise1.jpg**, we have performed: Opening followed by Closing.

Figure 1: res_noise1.jpg

For image **res_noise2.jpg**, we have performed: Closing followed by Opening.

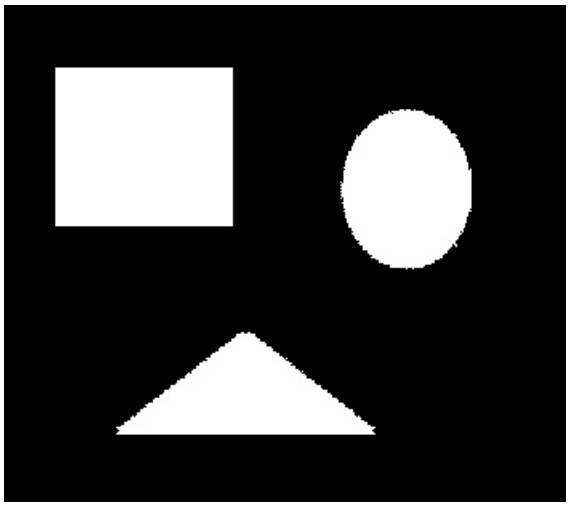

Figure 2: res_noise2.jpg

b) Please compare the two results, and indicate are they the same in your report. (1')

The results i.e res_noise2.jpg and res_noise2.jpg obtained from the above algorithms are the same.
Opening removes small objects from the foreground (usually taken as the bright pixels) of an image, placing them in the background, while closing removes small holes in the foreground, changing small islands of background into foreground.
Opening is the dual of closing, i.e. opening the foreground pixels with a particular structuring element is equivalent to closing the background pixels with the same element. Thus the images obtained by combining these operations are the same.

c) Using morphology image processing algorithms to extract the boundary 'res_noise1.jpg' and 'res_noise2.jpg', save the results and name as 'res_bound1.jpg' and 'res_bound2.jpg'. (2') p.s. Show all the results images and your answer in your report.

To find the boundary of the images we have performed the below steps:
1. Dilate the image
2. Erode the image
3. Subtract Erosion from Dilation

The images formed by the above operations on the images 'res_noise1.jpg' and 'res_noise2.jpg' are below:

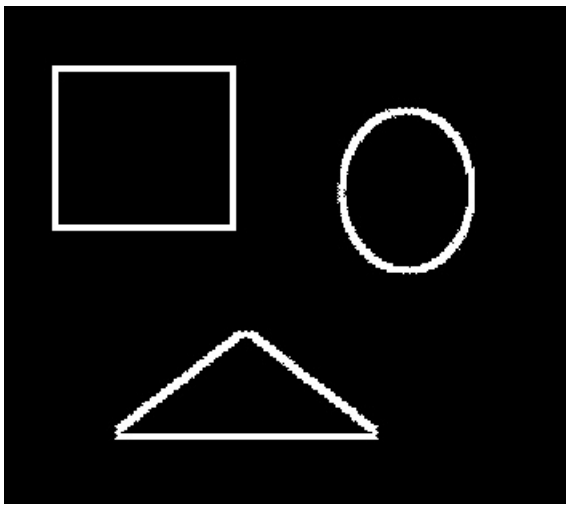

Figure 3: res_bound1.jpg

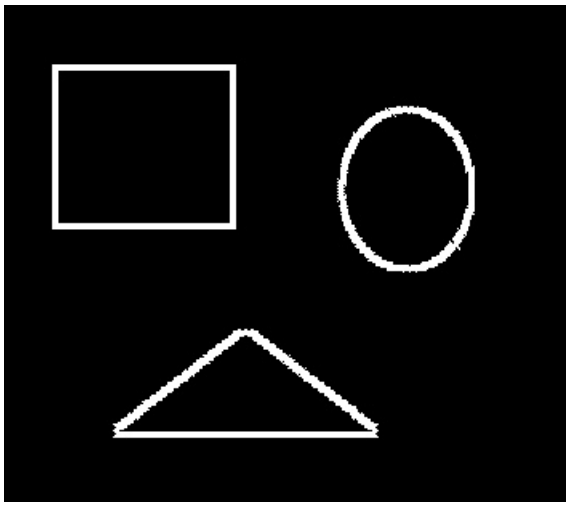

Figure 4: res_bound2.jpg

2 Image segmentation and point detection

a) The 'point.jpg' is a X-ray image of a turbine blade with a porosity. Using the point detection algorithm, you learned to detect the porosity. Label the points and give its (if more than one their) coordinates on the image. (2')

We have used a separate image than what is mentioned in the project document as the given image is not providing correct results. Link for the image used:
http://sse.tongji.edu.cn/linzhang/DIP/demoCodes/ch8/pointDetec/turbine-blade.jpg

We have used the below steps to detect the brightest point in the image:
1. The input image is represented as a set of discrete pixels, we have used the below convolution kernel that can approximate the second derivatives on the image.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

The center of the kernel has a value of 8 which is much greater than that of the other values which is -1. Thus any pixel which is brighter than the other pixels in the image, will be easily detected by applying convolution.

2. We have normalized the image to bring the values of the pixels in the same scale and then applied threshold on the image with a value of 120. The brightest point in now visible while the other noises have disappeared. The detected point has an intensity of 113 and has coordinates at **(249, 445)**.
The final detected point is shown in the below image:

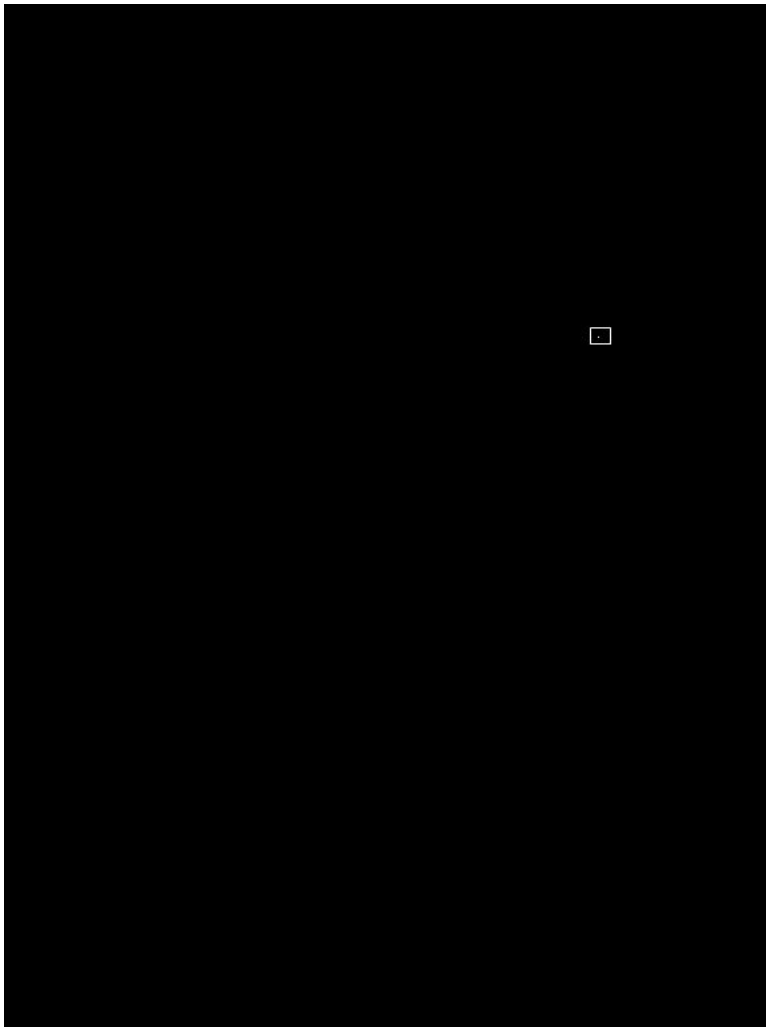

Figure 5: masked.jpg

b) Choose an optimal threshold to segment the object from background by thresholding on 'segment.jpg'. Show the segmentation results, and draw a rectangle bounding box on the segmented results (bounding box is the minimum box that covers all of pixels of the object within it, an example (bounding box) is shown below for you to refer). Give the coordination's for four corner points of the bounding box you drew. (3')

We have solved this problem by 2 approaches:

Approach1: Historgram:

Step1: We have solved this problem by implementing a historgram which will plot the count of every pixel along the y axis and the intensities along the x axis.

Step2: We can observe that the highest count in the histogram image is that of the chicken fillet and the value dropped drastically at around 200 with the brightest pixels, which can be considered a threshold as we are only concerned with identifying the bones.

Step3: We have applied threshold on the image with this point to detect the bones.

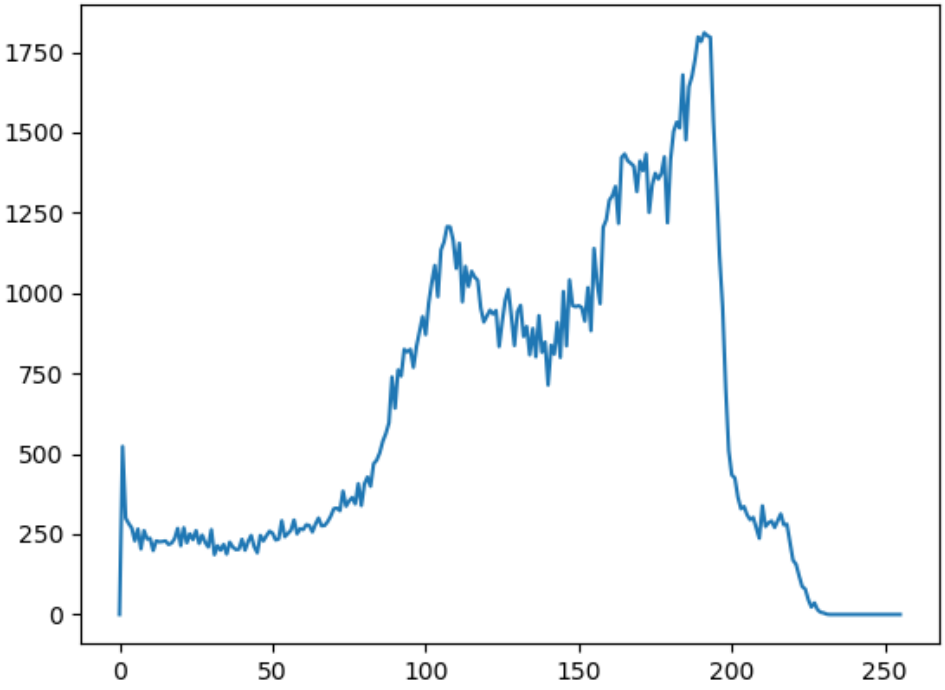

Figure 6: Histogram

Approach2: Heuristic Approach:

We have performed the below steps to detect the point in the image:

Step1: Discard the intensities with highest count as we can intuitively figure out that background chicken fillet must have the highest count. Choose pixels with intensities greater than 190.

Step2: Between 190 and 255, select an arbitrary threshold point t_init = 230 as initial threshold

Step3: Segment the image using T. This will produce two groups of pixels: G1 consisting of all pixels with gray level values ¿ T and G2 consisting of pixels with gray level values ¡= T

Step4: Compute the average gray level values μ_1 and μ_2 for the pixels in regions G1 and G2

Step5: Compute a new threshold value T = $0.5(\mu_1 + \mu_2)$

Step6: Repeat steps 3 through 5 until the difference in T in successive iterations is equal to 0.

Step7: After calculating, we apply the Global threshold obtained to the image to detect the chicken bones.

The image obtained by running the above mentioned algorithm is shown below:

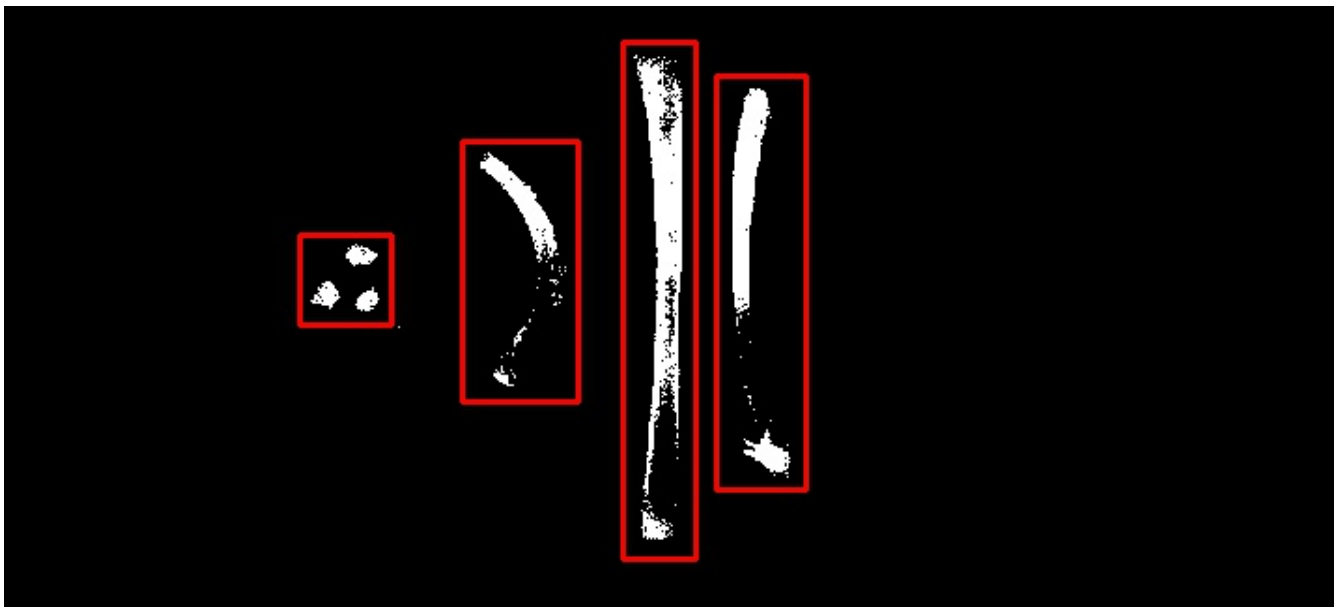

Figure 7: segment.jpg

We have drawn 4 bounding boxes that would wrap each of the chicken bones. The (x,y) coordinates of the four corners of the 4 bounding boxes from left to right is given by the table below:

Segment	**Top Left**	**Top Right**	**Bottom Left**	**Bottom Right**
Bone 1	381, 37	381, 258	429, 37	429, 258
Bone 2	331,19	331,295	370, 19	370,295
Bone 3	245,72	245, 211	307, 72	307,211
Bone 4	158,122	158, 170	207, 122	207,170

3 Hough transform [5'+3']

a) Utilizing the Hough transformation, design and implement an algorithm to detect all vertical (red lines), save the result image name as 'red_line.jpg' and indicate how many red lines you have detected. (2')

b) Detect all the diagonal (blue lines), save the result image as 'blue_lines.jpg' and indicate how many blue lines you detected. (3')

Hough Transformation : Lines The Hough transform is a feature extraction technique which detects imperfect instances of objects within a certain class of shapes by a voting procedure, allows us to figure out prominent lines in the image.

Parameter space representation A line in the image corresponds to a point in Hough space. It can be represented by the image below:

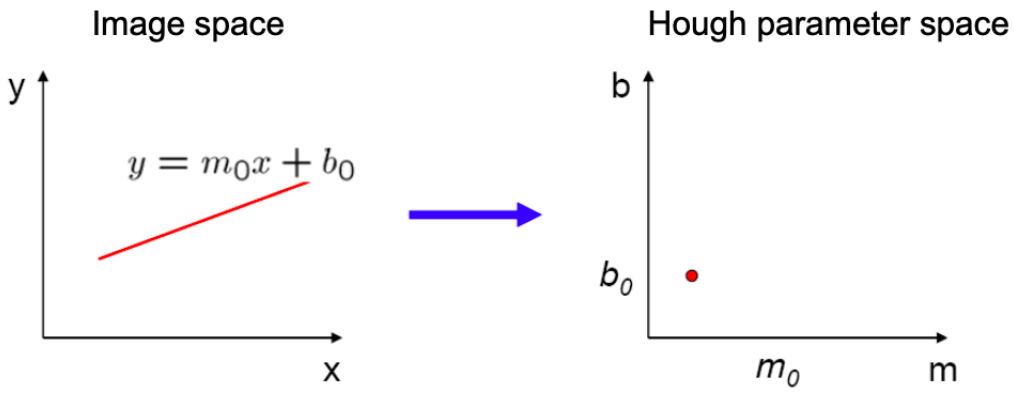

Figure 8: hough.png

Polar Representation: Each point will add a sinusoidal in the (θ,p) parameter space.

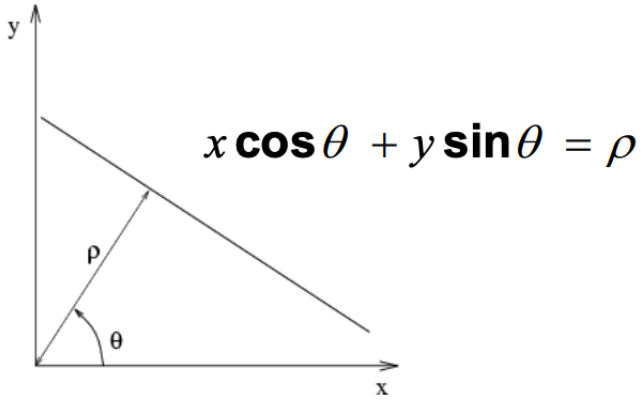

Figure 9: hough.png

We followed the algorithm below to detect lines using Hough transform:

Step1: Initialize accumulator H to all zeros.

Step2: Loop through every pixel of the edge detected image and if a pixel is zero, you ignore it. It's not an edge, so it can't be a line.

Step3: If a pixel is nonzero, generate its sinusoidal curve (in the pθ space) i.e take θ = -90 and calculate the corresponding p value. Then "vote" in the accumulator cell (θ, p).

Step4: Then take the next θ value and calculate the next p value. And so on till θ = +90

After executing the above algorithm on hough.jpg, we get the accumulator as below:

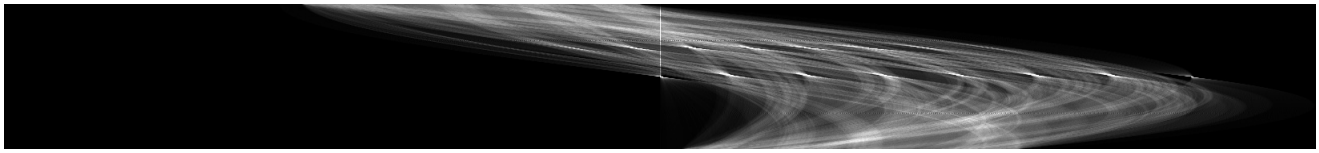

Figure 10: accumulator_line.jpg

The pixels with the greatest intensity will earn the most votes, we extract the peaks and convert it back to the x,y space. The resulting images for detecting the vertical lines(**6 lines detected**) and the slanting lines(**7 lines detected**) in the hough.jpg are as below:

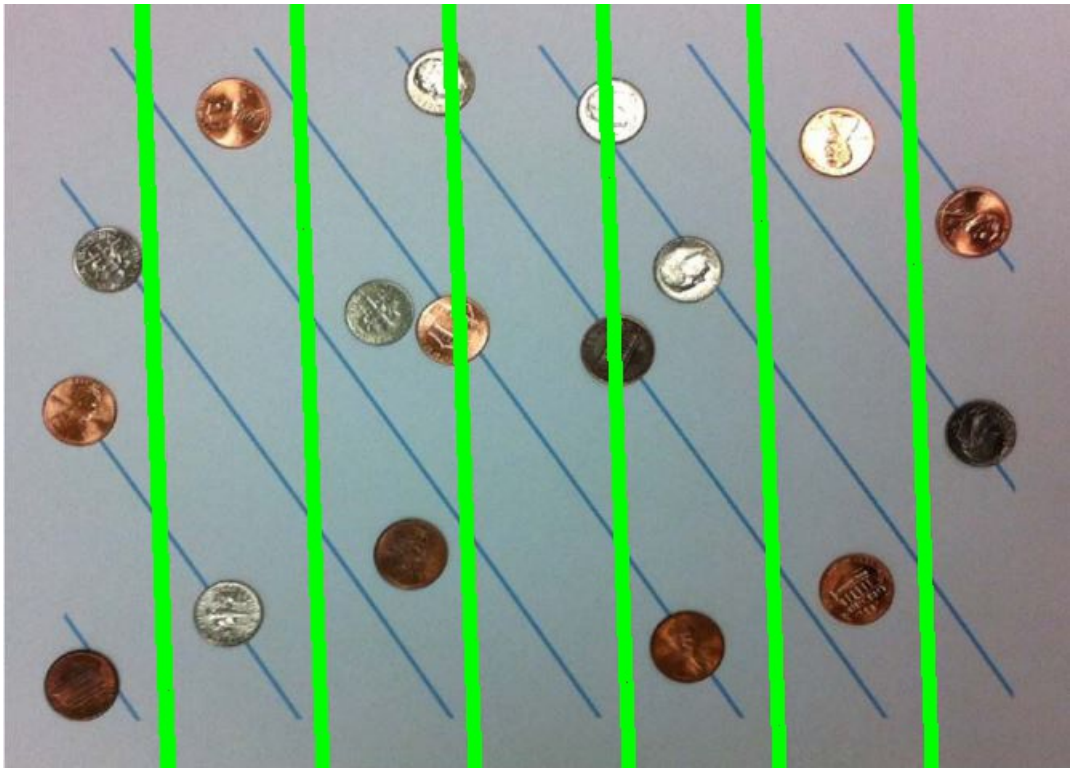

Figure 11: red_line.jpg

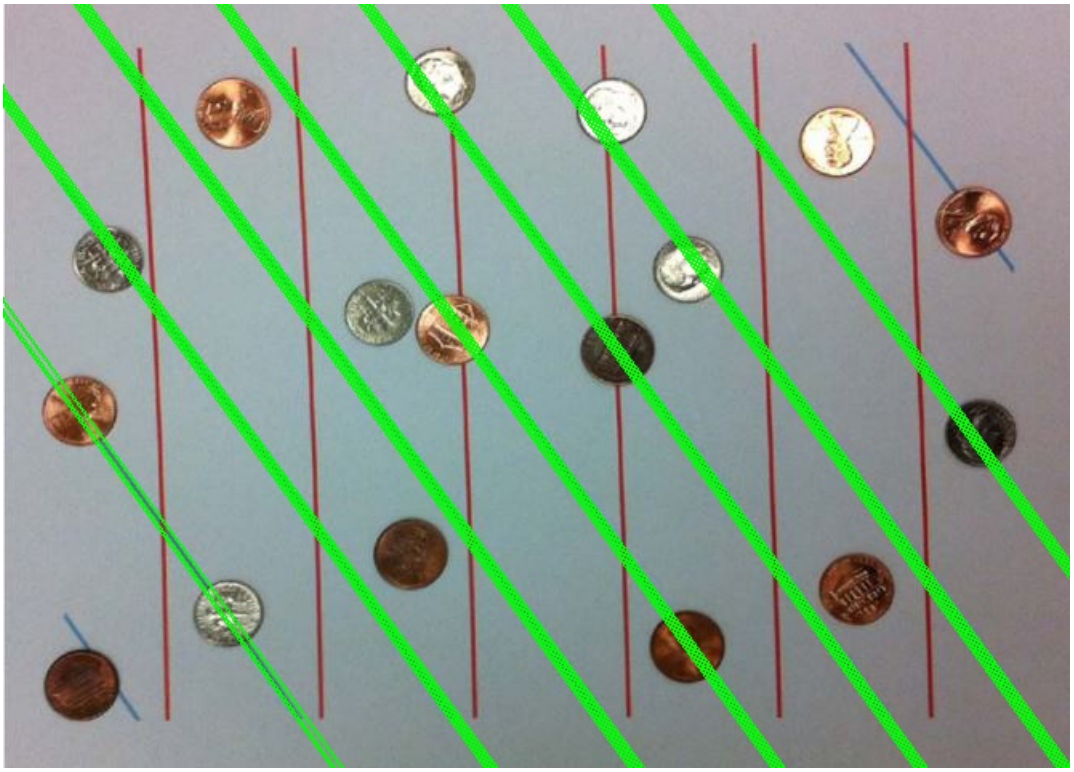

Figure 12: blue_lines.jpg

c) Detect all the coins in the image, save the result image as 'coin.jpg' and indicated how many coins you have detected. (bonus 3')

Parameterization: A circle can be described completely with three pieces of information: the center (a, b) and the radius. (The center consists of two parts, hence a total of three)

$x = a + R\cos\theta$

y = b + Rsinθ

When the θ varies from 0 to 360, a complete circle of radius R is generated.
In this case we assumed that R(=20) is known, thus the equations for a and b will be as below:

a = x1 - Rcosθ

b = y1 - Rsinθ

for a particular point (x1, y1). And θ sweeps from 0 to 360 degrees.

We followed the algorithm below to detect circles using Hough transform:

Step1: Detect edges and generate a binary image

Step2: For every 'edge' pixel, generate a circle in the ab space

Step3: For every point on the circle in the ab space, cast 'votes' in the accumulator cells

Step4: The cells with greater number of votes are the centers

After executing the above algorithm on hough.jpg, we get the accumulator for circle as below:

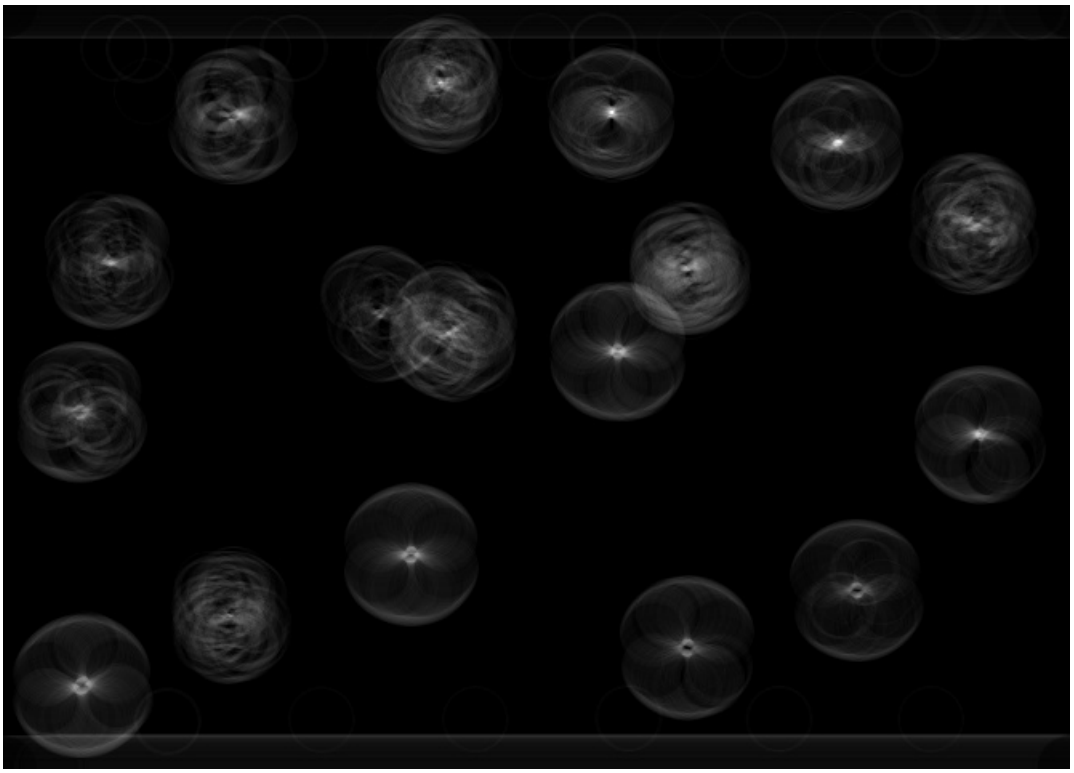

Figure 13: accumulator_circle.jpg

The pixels with the greatest intensity will earn the most votes, we extract the peaks and convert it back to the x,y space. The resulting images for detecting the circles(**17 coins detected**) is given below:

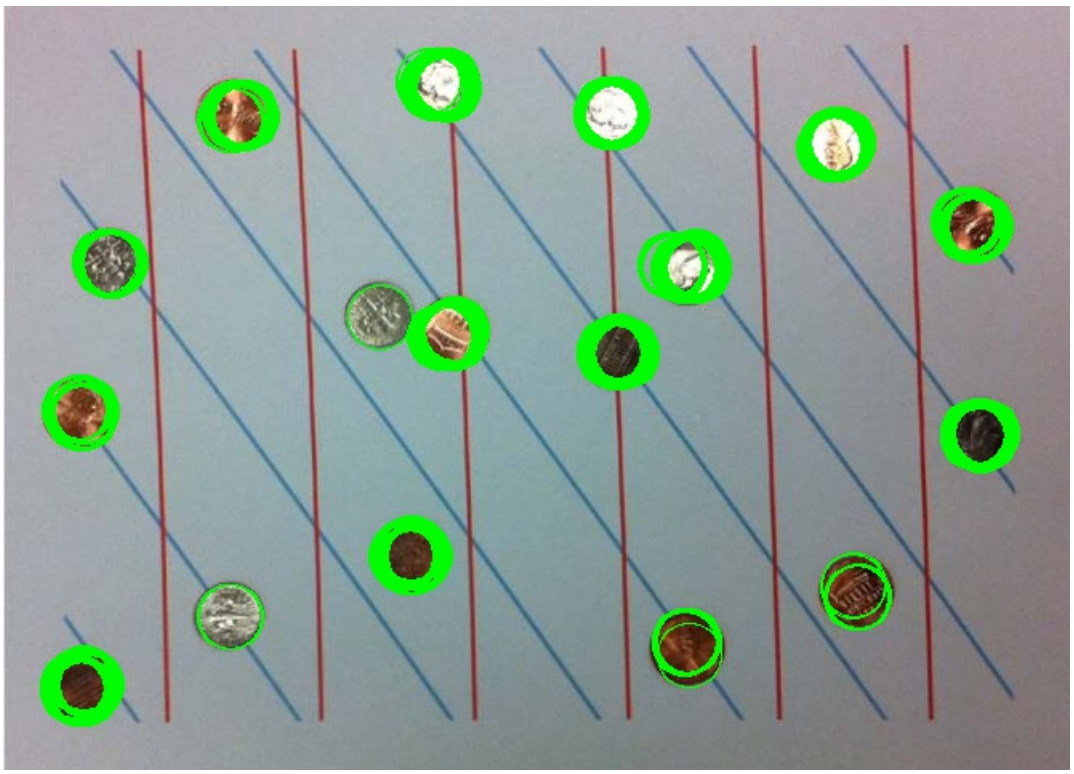

Figure 14: coin.jpg

4 References

The resources that has helped me understand the concepts of Morphology, Segmentation and Hough Transform are as follow:

- Hough Transform : http://www.aishack.in
- Hough Transform : https://en.wikipedia.org/wiki/Hough_transform
- Hough Transform : http://www.cs.unc.edu/ lazebnik/spring09/lec09_hough.pdf
- Introduction to Computer Vision: https://www.udacity.com/course/introduction-to-computer-vision–ud810