CHARU SINGH

2773417243

EE569

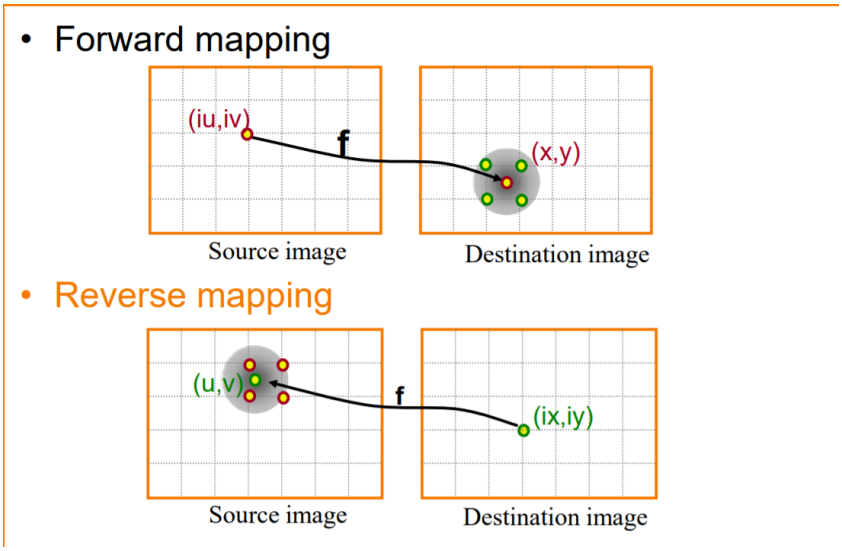
HOMEWORK 3

Image Warping:

Abstract:

Warping means that the pixels points are being mapped to points in order to obtain other shapes without any change in colors. It can be reconstructed only if the function is injective.

We have three main functions involved that is translation, scaling and rotation.



Procedure: we have used inverse mapping here

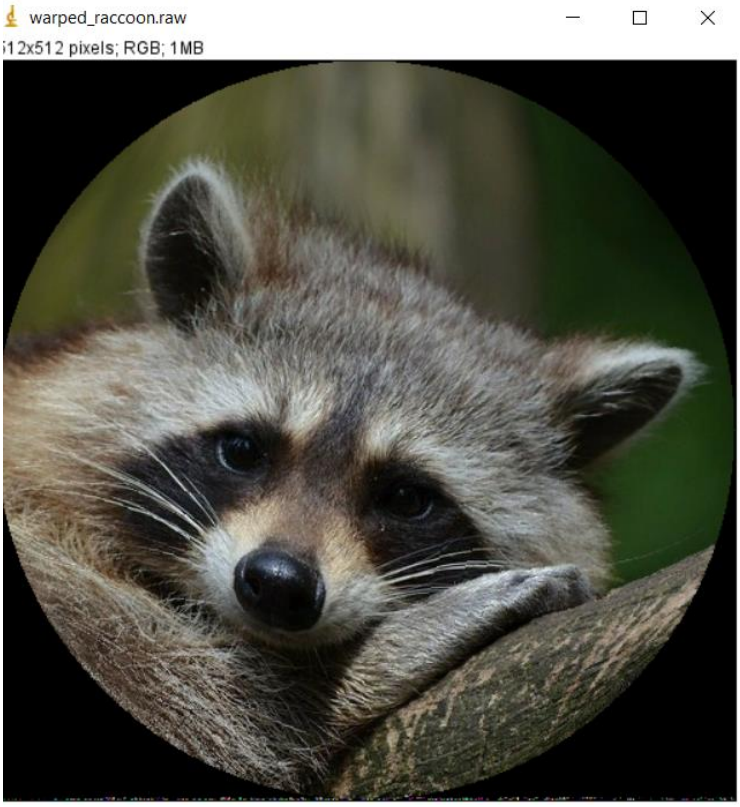
1. First convert the coordinates from image to cartesian using following formula:
   xk=1+column-0.5
   yk = Height of image +0.5-rows
2. Normalize xk and yk by using :
   xk_normalized=(xk-255)/255;
3. Mapping function will be :
   For square to circle:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x\sqrt{1-\frac{y^2}{2}} \\ y\sqrt{1-\frac{x^2}{2}} \end{bmatrix}$$

   For circle to square:
   x'=x/sqrt(1-(y^2/2))
   y'=y/sqrt(1-(x^2/2))

4. Then again denormalize above values.
5. Convert from cartesian to image coordinates.
6. Floor the above coordinates.
7. Apply bilinear interpolation in order to get artifacts.

Results:



warped_bb8.raw

512x512 pixels; RGB; 1MB

warped_hed.raw

12x512 pixels; RGB; 1MB

Homography:

Motivation:

In order to find out the corners of an image we have various algorithms for it because the corners will remain corners even if an image is rotated. But when we scale images like in panorama, those corners will be flat in a small window. As a result, Scalar Invariant Feature SIFT algorithm extracts some control points and find its descriptors.

SIFT Algorithm:

1 **Scale space Extrema Detection:**

In order to detect larger corners, we need large windows. With the help of various sigma values Laplacian of gaussian is found.

2. **Keypoint Localization:**

After finding the locations of keypoints ,taylor series expansion is used to extrema location and comparing with threshold we will decide accordingly.

3. Orientation Assignment:

Orientation is being assigned to every key point. invariance to image rotation is achieved. Around the keypoint location a neighborhood is taken, and direction and gradient are calculated.

4. **Keypoint Descriptor:**

Around a key point 16*16 patch is taken as a key point which is divided into 16 blocks of 4*4 size. Histogram is created for each sub block. All the bin values are represented as a vector in order to form descriptor.

5**. Keypoint Matching:**

Keypoints between the images are being matched by their nearest neighbors, second closest match is also considered sometime.

Algorithm:

1) Reading first image from image set

rimg(i,j) = rawim(k)
gimg(i,j) = rawim(k+1)
bimg(i,j) = rawim(k+2)
k = k+3;

2) → Convert to gray scale image

gray_mid = 0.2989 * rimg + 0.5870 * gimg + 0.1140 * bimg;

3) Convert to binary image.

gray_mid | 255;

4)

5) Read left.raw image & right.raw same as above.

6) Using SIFT

points = detectSURFfeatures (gray_mid);

[features, points] = extractFeatures (gray_mid, points)

7) Store points & features

Prev_points = points;
Prev_features = features;

1500 2000

8) Detect & extract SURF features for I(n)

left_point = detectSURFfeatures (gray_left)

[features_left, points_left] =

extractFeatures (gray_left, points_left)

9) Find correspondances b/w I(n) & I(n-1)

b) find h matrix

h_left = $\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & x_8 \\ 1 & 1 & 1 & 1 \end{bmatrix}$

Then find h_right

H matrix obtained is

Hleft = $\begin{bmatrix} 1.0314 & 0.0461 & -20.2400 \\ -0.2235 & 1.0229 & 231.7971 \\ 0 & 0 & 1 \end{bmatrix}$

HRight = $\begin{bmatrix} 0.9752 & -0.0396 & 23.1261 \\ 0.0388 & 0.9727 & -162.0811 \\ 0 & 0 & 1 \end{bmatrix}$

Morphological Image Processing:

Motivation: In application such as object recognition representation of a shape plays a vital role in image processing. The thinning process gives connected skeleton having spurious branches which can be overcome by pruning technique. It is most commonly used on binary images.

Abstract:

To ensure the connectivity the, total erasure is prevented by controlling erosion process.

Shrinking: our goal is to erase 0 intensity pixels such that an image with without holes will shrink to a single pixel near its center of mass and an image with holes will shrink to a connected ring.

Here we are performing three morphological operations that is Shrinking, thinning and skeletonizing. In order to remove the foreground pixels from the images such as opening or erosion we use **Thinning.** This operation involves hit and miss transform.

Other operation is known as **skeletonizing** whose aim is to represent general form of an object by extracting a feature (region-based shape). For each point in an object Medial Axis transform will determine the closest boundary points. If the inner point contains at least 2 boundary points, then it belongs to the skeleton.

Key idea:

We use structuring element which is overlapped on all locations in an image and is being compare to neighboring pixels. Some of the applications demands for the

element to get fit within the neighbor while some of them find out whether it hit or miss the neighborhood.

Results:

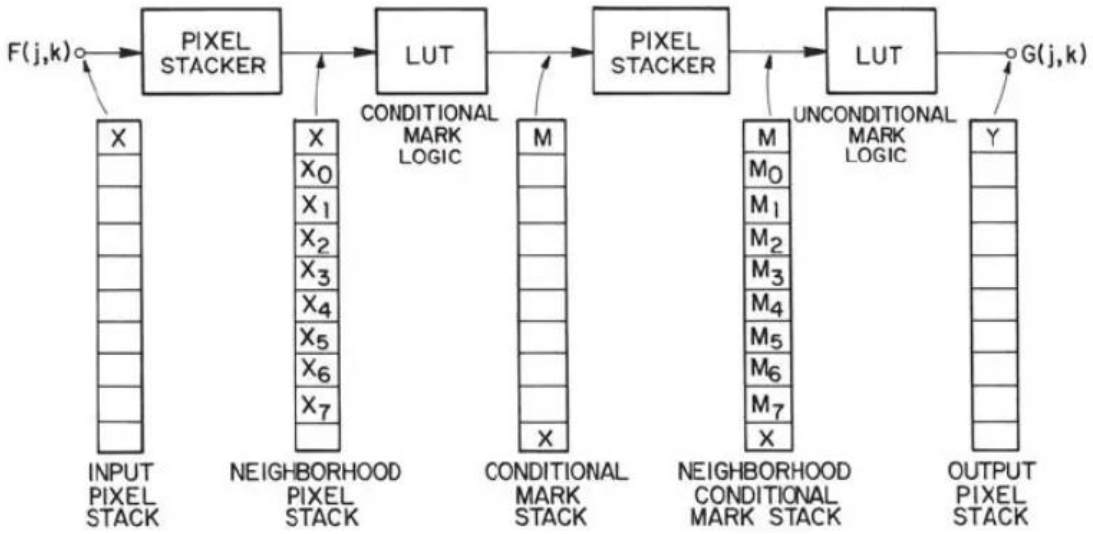
In thinning we can observe that there are spurious spurs in the output whereas a skeleton gives a compact representation that preserves topological characteristics of original image

Procedure for shrinking, thinning and skeletonizing:

1. First convert gray scale image to binary image and name it as X
2. Apply shrinking algorithm using the pattern table for shrinking. First apply the conditional mask and get M matrix using the following procedure: First check whether the center pixel is 1. If yes, then compare the 3*3 image pixels for corresponding i,j with the given pattern table. If it matches with any one of the patterns assign that pixel value to one.
3. Now we have a matrix M, now using this matrix as input find a new matrix P (initialized to zeros) using the unconditional mask pattern table. Repeat the procedure as mentioned in point 2.
4. Now we have P as a marked element.
5. Calculate final image G by using following formula:

$$G(j,k) = X \cap [\overline{M} \cup P(M, M_0, \ldots, M_7)]$$

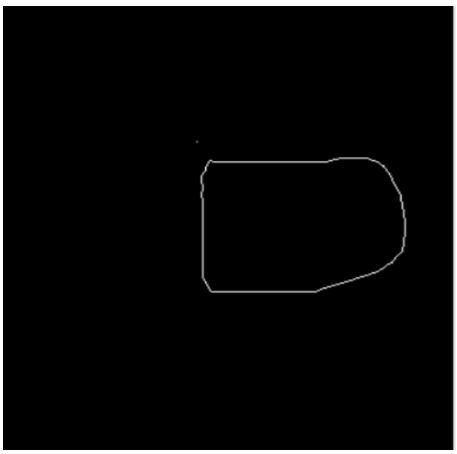
6. Now keep on iterating until you get the saturated output that is no more variation in the output image.



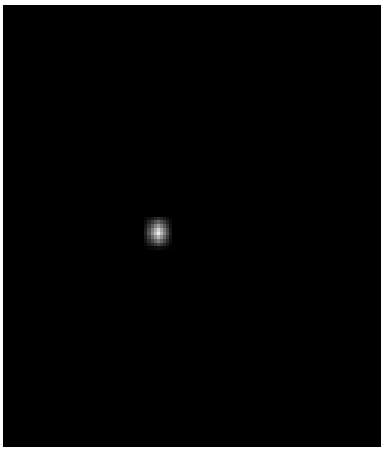
This is a flowchart for conditional mark operations

Results:

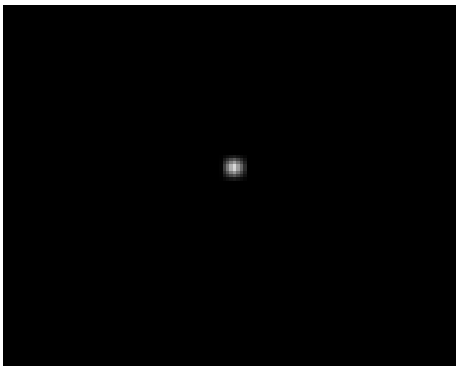
Shrinking:



Cup

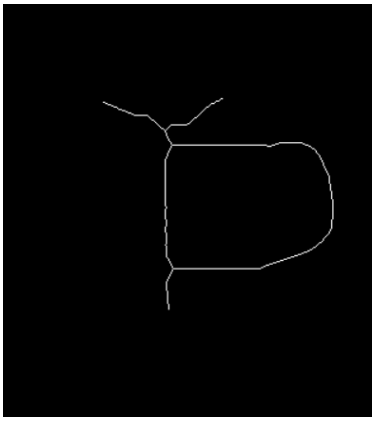


Fan

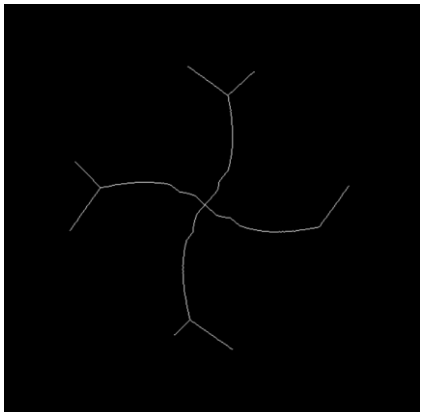


Maze

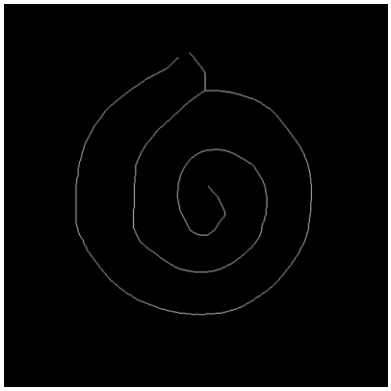
1. Thining:



Cup

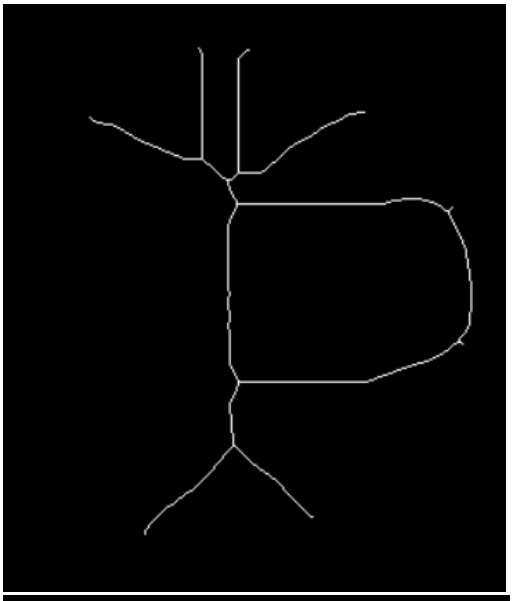


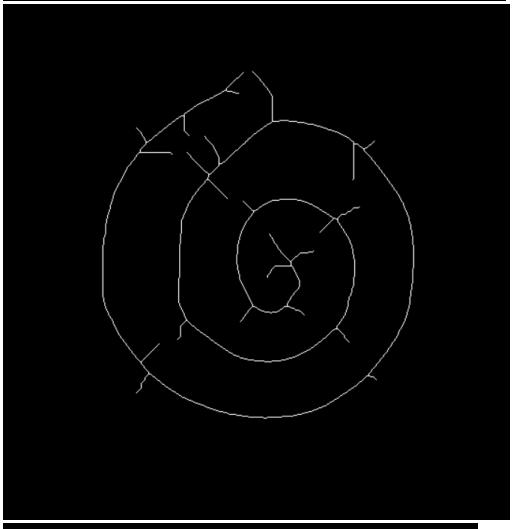
Fan

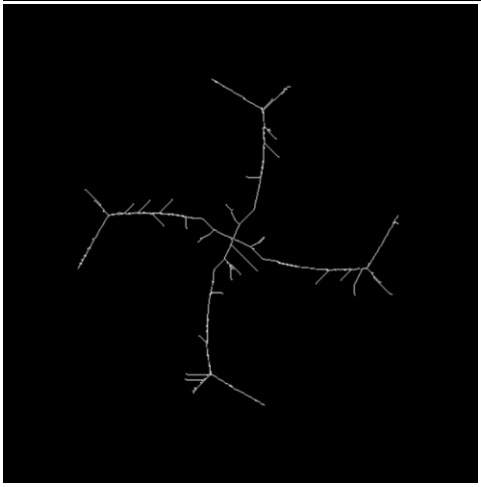


Maze

Skeletonizing:







Observations:

In shrinking we will get all white pixels converge to a single dot. In cup image we will get a circle and a dot because of the ring formation.

In thinning we will get the boundary the overall structure of an image

In skeletonizing you can observe that there are branches alond radial axis of an object.

Procedure for star problem:

Method 1:

1. In order to find the number of stars in an image first apply shrinking to algorithm and then count number of pixels with intensity values 1 or 255 in an image.
2. In order to find the size of each stars and their frequencies you need to perform **connected component labelling**.
   Connected Component Labelling:
   a. First pass: first scan the whole image and apply the operation only when the center pixel is 1(Raster Scanning).
   b. Now for a pixel see its neighboring elements if there are no neighbor with the labels, assign a new label to current element.
   c. If there are neighbors with the labels, then assign minimum label to that pixel.
   d. Update the equivalence table in which we have assigned labels to our pixel coordinates.
   e. Second pass: update your equivalence table that is for every encounter with different labels in the neighbors, mark them all as one label only meaning that they are connected.
   f. Now finally your image will get divided into various labels giving us how many connected components are present in an image.
3. Now in order to find out star size count number of labels and corresponding to that find number of pixel value 1 assigned to that particular label.

RESULTS:

Conclusion:

There are 112 stars if we apply shrinking algorithm to the image.

There are 113 stars present in an image. I used connected component labeling method in order to find number of stars and frequency of each star size is calculated in code itself. There are 8 different star sizes. Below shown is the frequency of star sizes (right column).

48 1

41 1

66 1

19 1

40 1

32 1

44 2

43 1

129 1

28 1

23 1

4 1

38 2

84 1

33 1

5 2

22 3

13 10

27 1

30 2

54 1

7 2

152 1

11 6

34 1

14 2

25 2

96 1

49 2

16 3

12 8

10 8

20 6

21 1

115 1

17 6

15 7

9 9

8 7

6 1

18 1

24 2

Procedure for PCB analysis:

1. Perform shrinking operation on the image you will get some circles and lines as output.
2. Then apply the structuring element on whole image and check the pixel neighbor only if its value is one, whether they are zero or not. To count number of holes, consider only those pixels whose all 8 neighbors are zero.
3. Set the counter to count number of holes.
4. In order to find number of paths you must apply connected component labelling method as mentioned above. Remember to invert the image and make paths as white.
5. Now you will get total number of connected components as different labels.

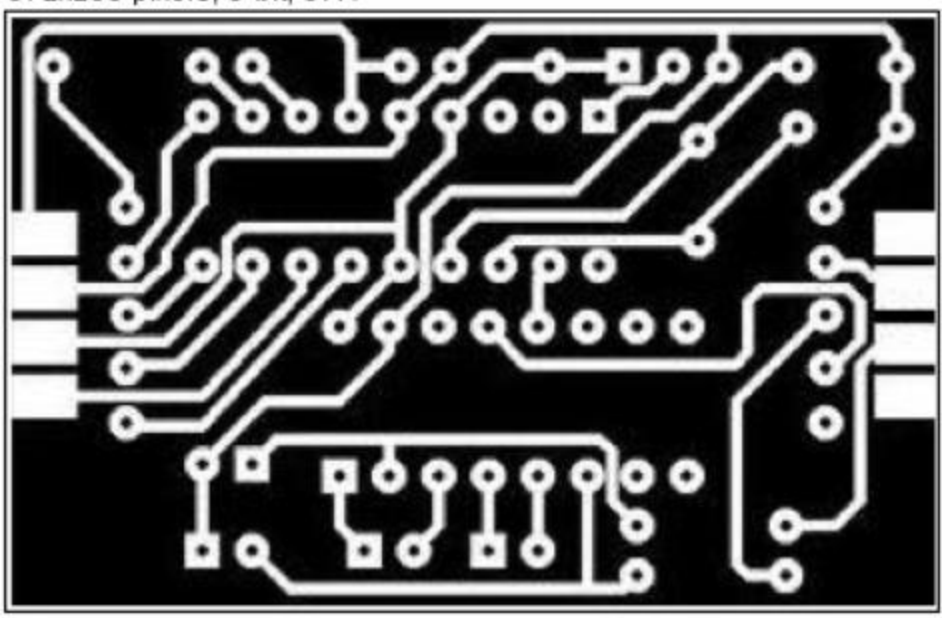
Results:

There are 72 holes present in the image which is calculated by applying shrinking algorithm.

There are 25 unique paths in the image which are calculated using Connected Component labelling.

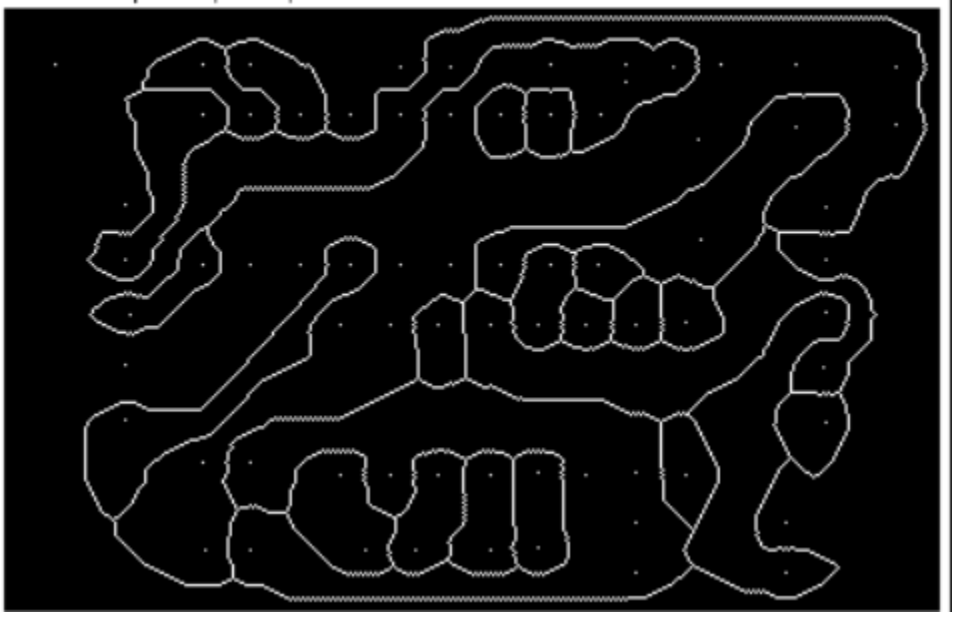
inverted_PCB.raw

372x239 pixels; 8-bit; 87K



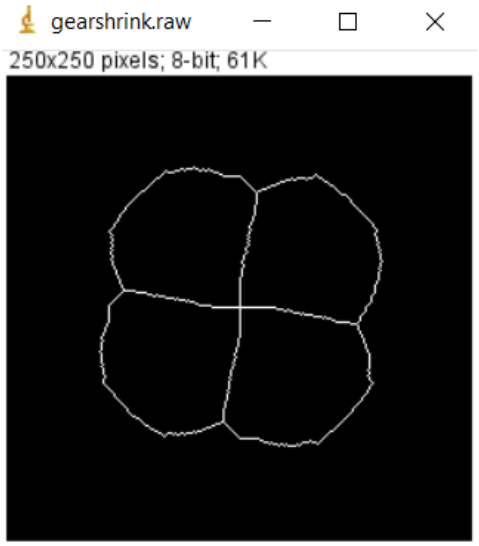
pcboutput.raw

372x239 pixels; 8-bit; 87K



Gear:

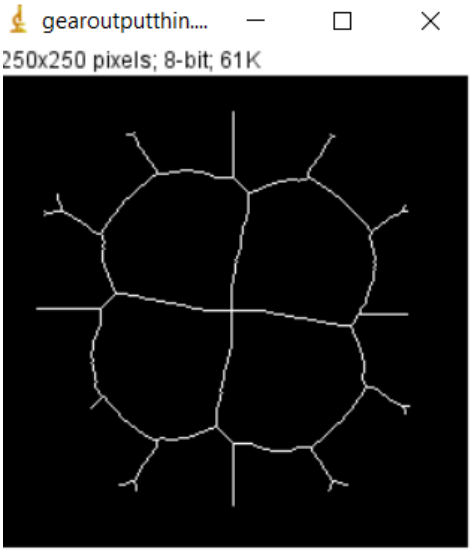
Results:



Output after applying shrinking.

Center of outer circle is located at 124,125.

Radius: 19 if found by filling all the holes and then starting from center check until black pixel arrives.



Output after apply thinning.

Procedure:

1. Find the center of the outer circle. And radius as well.
2. There should be 12 teeth but two are missing.
3. Rotate every 360/12=30 degree and check the radius if its greater than 19 then tooth is present else not.
4. Note the location of the pixel at which the tooth is missing.

References:

[1].MATLAB Panorama example: https://www.mathworks.com/help/vision/examples/featurebased-panoramic-image-stitching.html?searchHighlight=stitching&s_tid=doc_srchtitle

[2] OPENCV feature matching example: https://docs.opencv.org/3.4/d5/d6f/tutorial_feature_flann_matcher.html

[3] lunch rooms from "PASSTA Datasets", Cvl.isy.liu.se, 2016. Available: http://www.cvl.isy.liu.se/en/research/datasets/passta/.

[4] MPEG-7 Shape dataset: http://www.dabi.temple.edu/~shape/MPEG7/dataset.html

[5] http://www.inf.u-szeged.hu/~palagyi/skel/skel.html