EE569: Homework #3

Issued: 02/17/2020 Due: 11:59PM, 03/03/2020

Problem 1: Geometric Image Modification

(a)Geometric warping

Geometric Wrapping is a transformation in which the coordinates of the pixel are changed and not the intensity. Thus, it produces images with different effects. In the given question, the image must be wrapped from a square to a disk. The wrapping technique used is Elliptical Grid Mapping. The center, and the boundary pixels are mapped to the corresponding boundary and center pixels in the wrapped image. It is one to one mapping.

Procedure

1)The image coordinates (x,y) are converted to the cartesian coordinates (xcart,ycart)

2)The mapping is done from a unit square to unit circle (everything should lie between -1 and 1). Therefore, modify the pixel coordinates to lie between -1 to 1.

xcart=(float)(x-(Width/2))/(float)((Width/2))

ycart=(float)((Height-y)-(Height/2))/(float)(Height/2);

3)The mapping from square to disc is done using the following equation

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

4)These coordinates are converted back to the image coordinates and the wrapped image is displayed. Reverse the equation in step 2.

Original Image

Reversed Image

Racoon.raw

Wrapped Image Raccoon.raw

Reverse Image Racoon.raw

Original Image bb8.raw

Wrapped Image

Reverse Image

b) To reverse the image:

1)The (u,v) points on the disc are mapped to the square (x,y) using the following formula

$$x = \frac{1}{2}\sqrt{2 + u^2 - v^2 + 2\sqrt{2}\,u} - \frac{1}{2}\sqrt{2 + u^2 - v^2 - 2\sqrt{2}\,u}$$
$$y = \frac{1}{2}\sqrt{2 - u^2 + v^2 + 2\sqrt{2}\,v} - \frac{1}{2}\sqrt{2 - u^2 + v^2 - 2\sqrt{2}\,v}$$

2)Inverse mapping is used to map points. The (u,v) values are the same as the ones that we obtained using the wraping stage. Using these values the (x,y) are obtained using reverse mapping. Round the values, because it can produce decimal values.

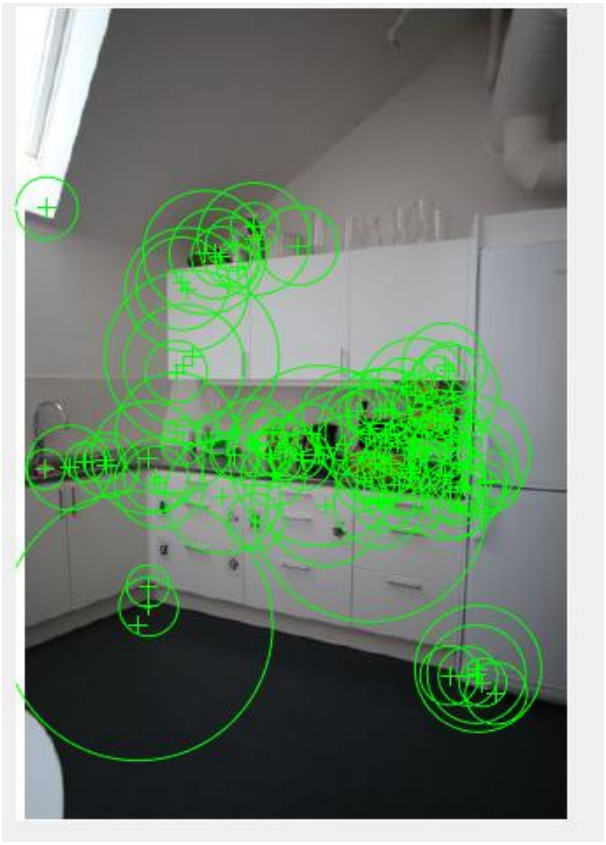
3) Again, these values are mapped to image coordinates and the reversed image is obtained.

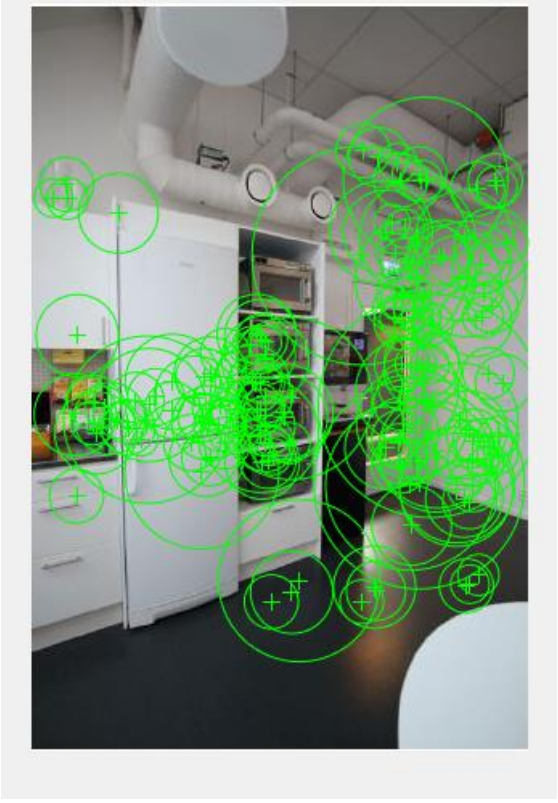
c) The Original Image and the Reversed Image are almost the same because only the coordinates are changed and not the intensities. The Reversed Image might have black dots if forward mapping is used Since we are using reverse mapping, we do not have those problems.

Homographic Transformation

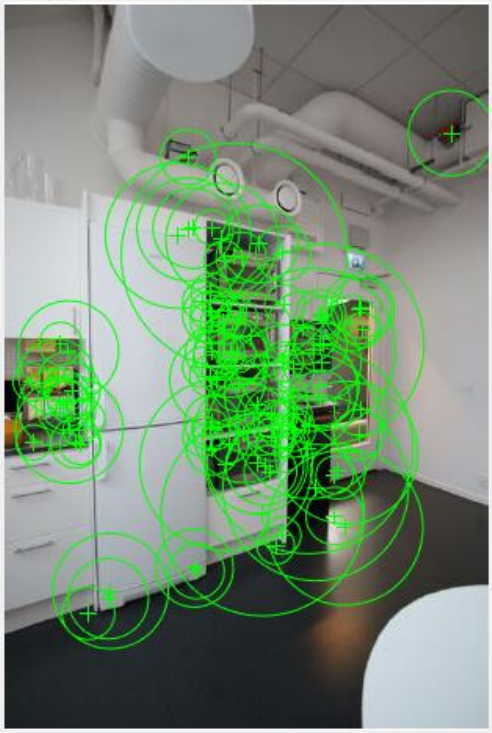
Left Image

Middle Image





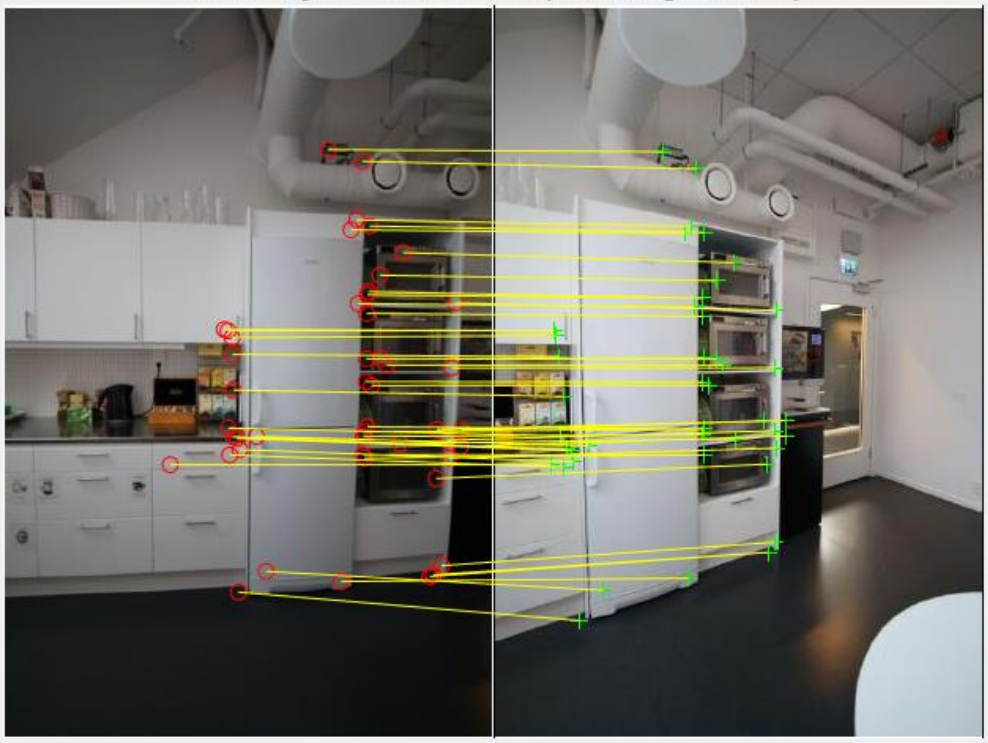
Finding strong points

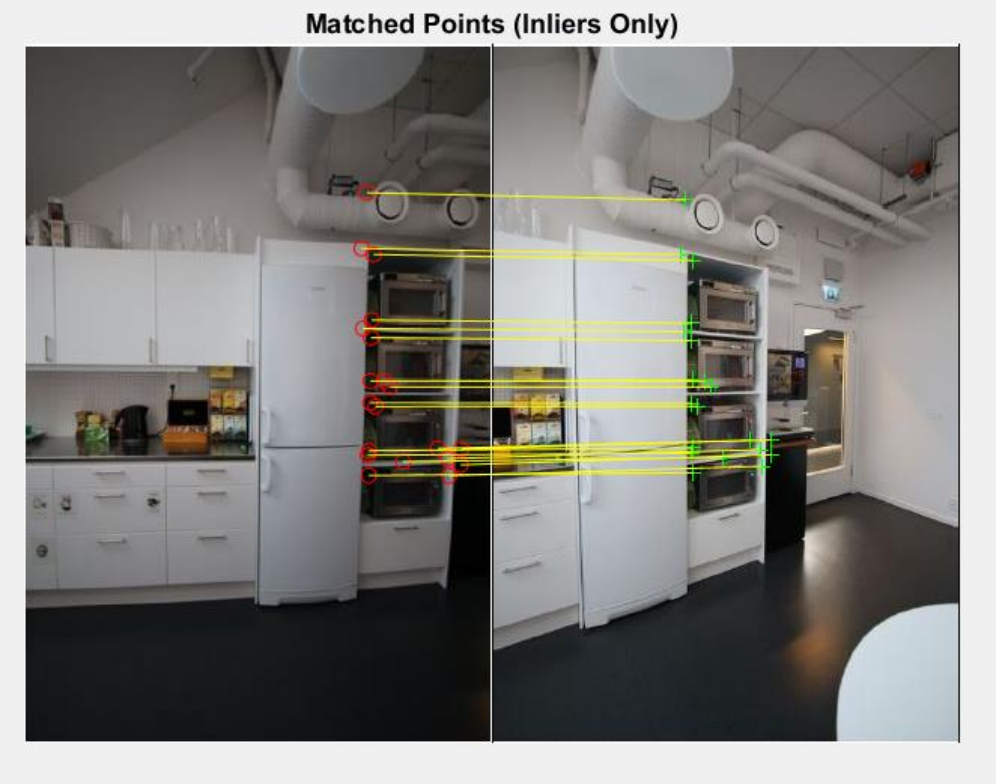


Putatively Matched Points (Including Outliers)

Matched Points (Inliers Only)

Putatively Matched Points (Including Outliers)





Problem 2: Morphological processing

Morphological Image processing has a lot of nonlinear image processing that deals with the form and the structure of the image. It is used in object selection, connectivity analysis.

(a)Basic morphological process implementation

Procedure

1)The image is traversed pixel by pixel. The input to the system should be binary image. The foreground is ‘1’ and the background is ‘0’. Hit and miss transformation is used.

2)If the center pixel in a neighborhood of 3x3 is a foreground pixel it is passed to first stage otherwise nothing is done to the pixel.

3)The 3x3 neighborhood of center pixel in the binary image is compared to the unconditional masks with S ‘Shrinking’ T for ‘thinning’ and ‘K’ for skeletonizing. If the pattern exactly matches, then that pixel is marked for further consideration.

3)Hit and Miss is a two-stage procedure. The output of the first stage is passed to the second stage.

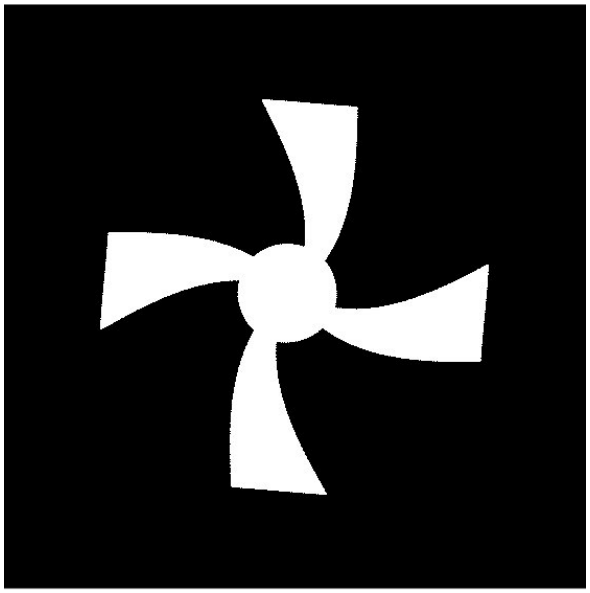
4)In the second stage only the marked pixels are considered. The 3x3 neighborhood pixels are compared to the conditional masks specific for Shrinking, Thinning and Skeletonizing. If there is an exact pattern match, then that pixel is selected.

5) The resulting pixel that location would follow this formula:

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

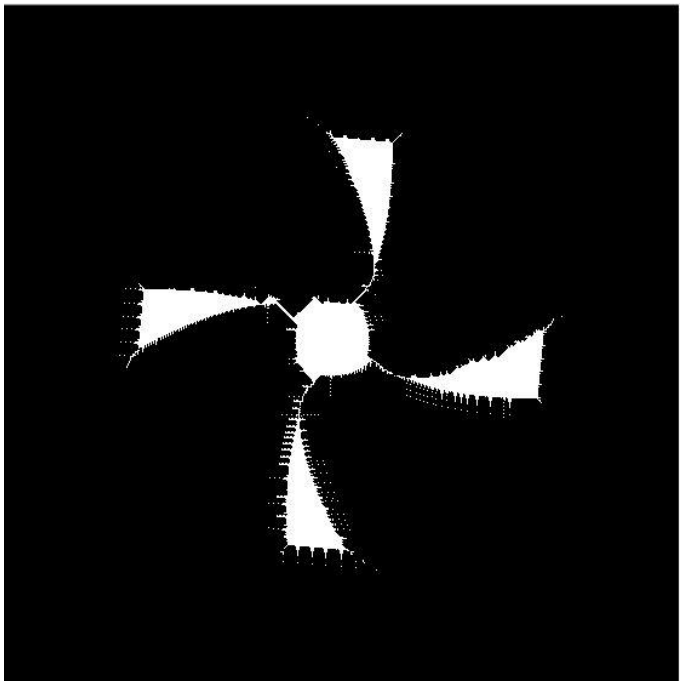
Where X is the input pixel, M is the M array from the First stage. P is the Second Stage Output. It determines whether the pixel must be erased or not. G(i,k) is the final output. This output image is fed as an input for the next iteration.

6) The steps are repeated for several iterations until there is no change in the input image and the outputimage.

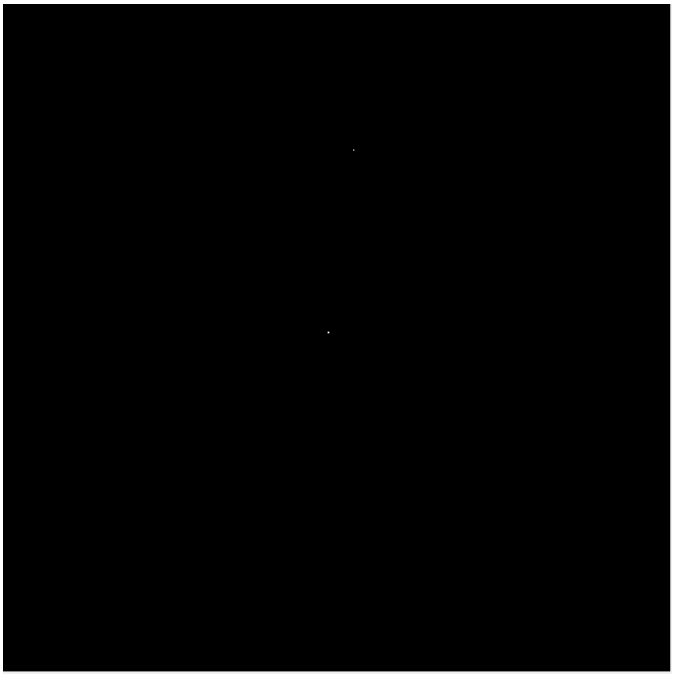


Original Image

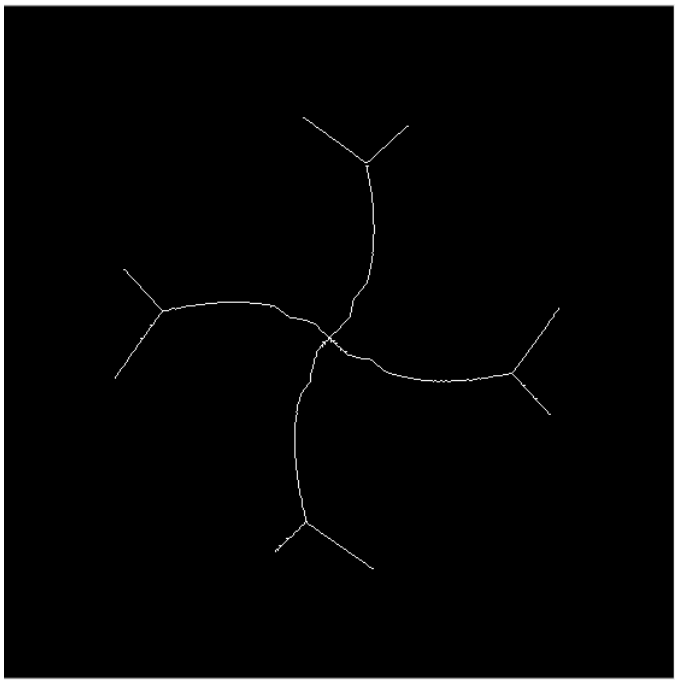
The convergence took 220 iterations for Shrinking. The convergence took 50 iterations for Thinning. The convergence took 36 iterations for Thinning.



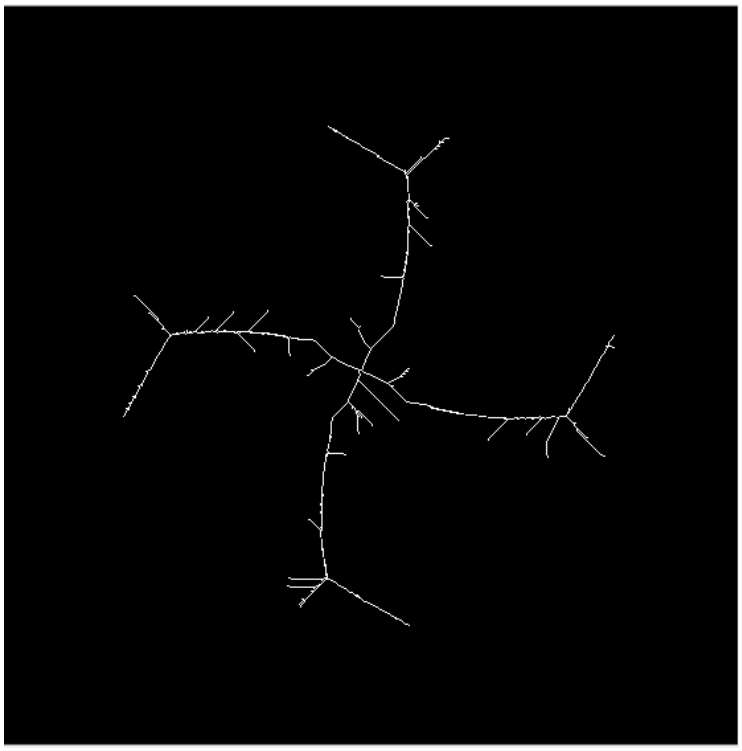
Intermediate stage of shrinking



Output of Shrinking Fan.raw

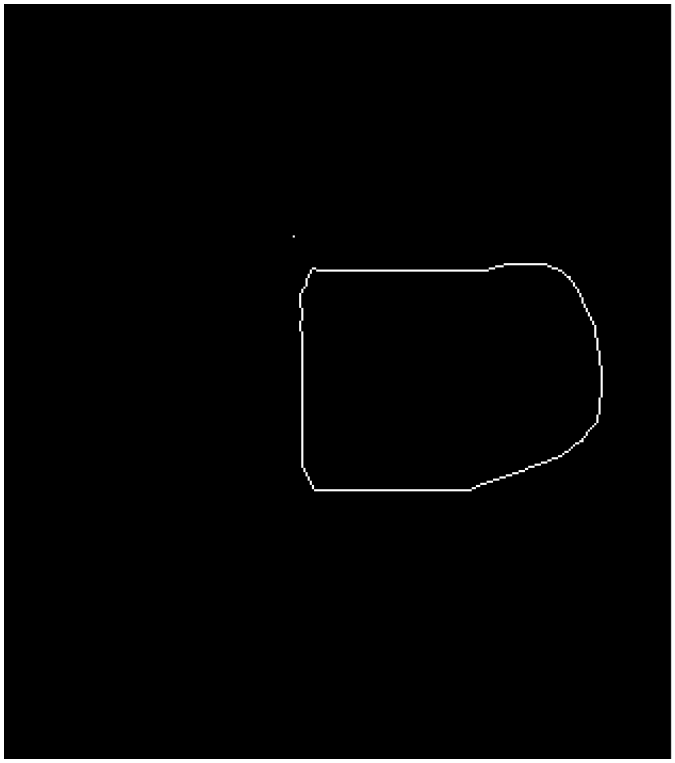


Output of Thinning Fan.raw

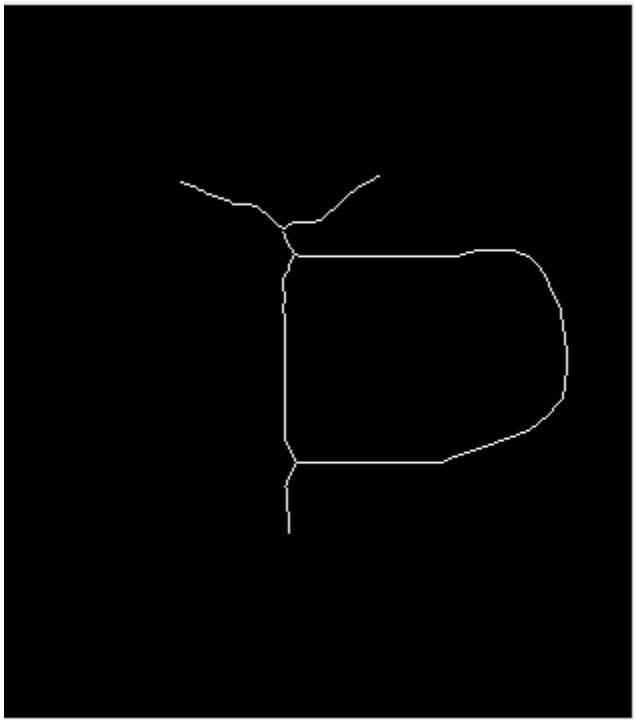


Output of Skelotonizing Fan.raw

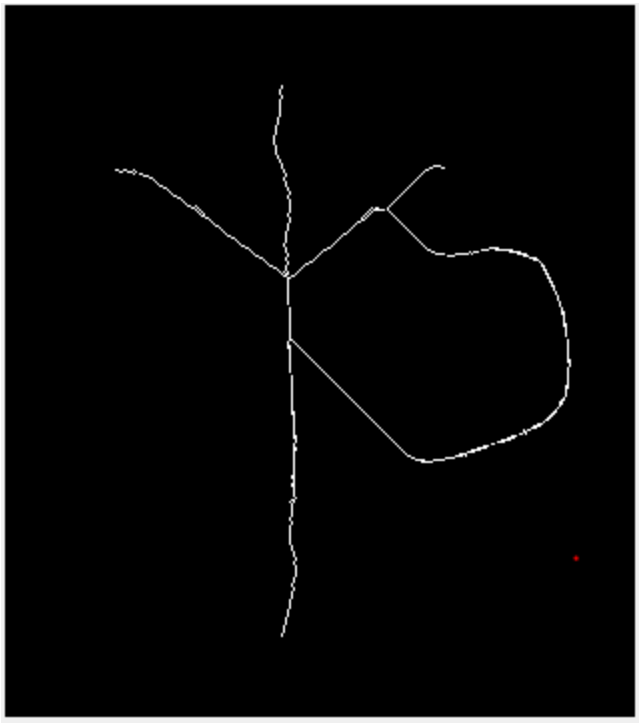
Input Image Cup.raw



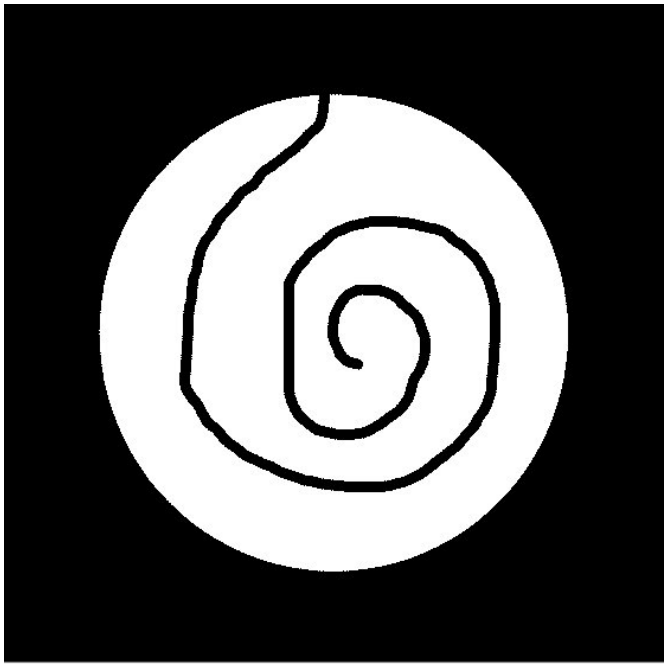
Shrinking output of Cup.raw



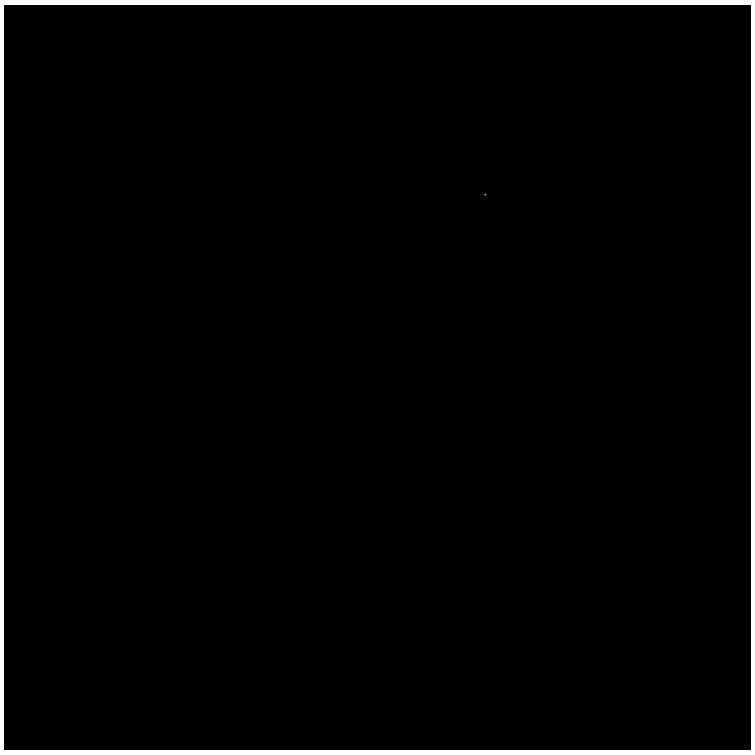
Thinning Output of Cup.raw(Convergence at 91 iteration)



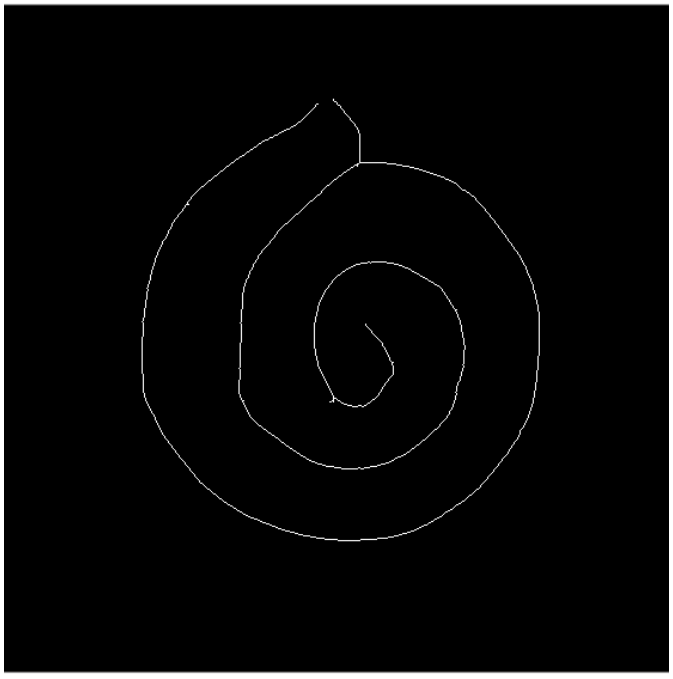
Output of Skeletonizing Cup.raw



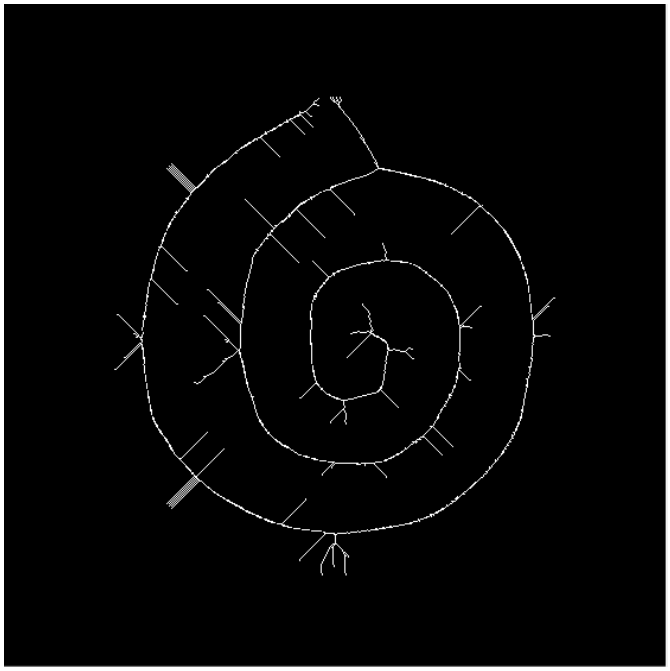
Original Image Maze.raw



Output of Shrinking Maze.raw



Output of Thinning Maze.raw (Convergence at 50 iteration)



Output of Skeletonizing Maze.raw

Discussion

Skeletonizing is used to extract the region-based shape features in order to obtain the general form of the image. Skeletonizing preserves more information and is sensitive to noise. It maintains the structure of the binary image. Shrinking is the strongest morphological process. It can reduce the binary image to a single dot as seen in fan.raw. Shrinking erases the pixels such that an object without holes erodes to a single point towards the center of mass. A two stage is used in order to get more information so that the correct candidates are chosen. Thinning is a process where object without holes are eroded to a minimally connected stroke and an object with holes forms a connected ring. This can be observed in the Cup.raw output for thinning. Thinning sometimes produces the same output for different objects. But Skeletonizing is unique.

(b) Counting games

The algorithm used to find the count of stars is connected component labelling. It is widely used to detect the connected regions in a binary image. It creates a labeled image in which all the same connected positions in a binary image will have a unique label. In this algorithm I have used 8 connectivity.

Procedure:

Convert the image to binary image

The algorithm uses two passes:

First pass:

- Check if the current pixel (i,j) is a foreground and not background
- If the center pixel is not background(0):

  1)Find the neighbouring elements of the current pixel (i,j).

i-1,j-1	i-1,j	i-1,j+1
i,j-1	i,j	

  2)If there are no neighbors give a unique label to the center pixel.

  3)If there are neighbors with labels, find the minimum among the neighbors and assign that label to the center pixel.

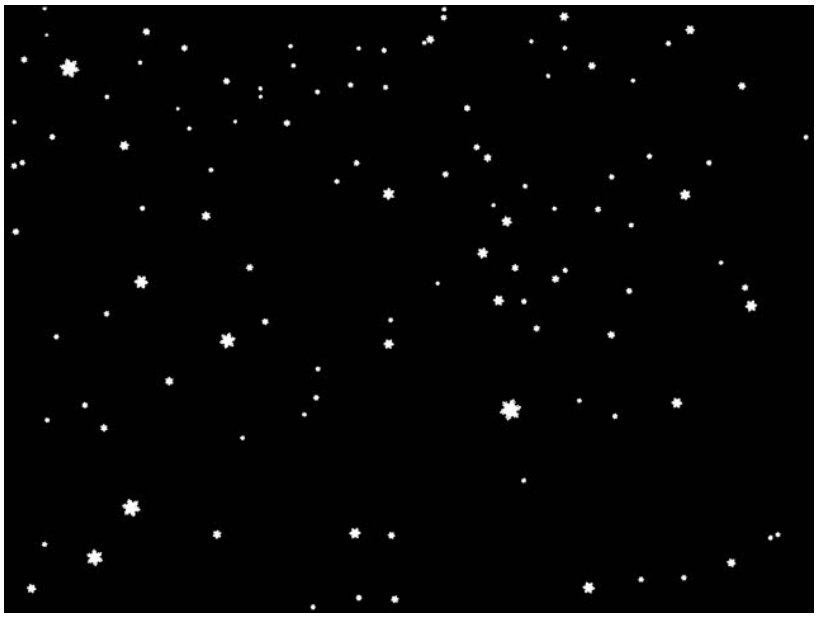
  4)If the neighbors have more than one label, then the equivalence array is created to store the equivalence between the neighboring pixels.

- Go to the next pixel and repeat the process

Second pass:

1)Traverse through each pixel in the image

2)Based on the values stored in the equivalence array, the labels are updated with the lowest equivalence label.

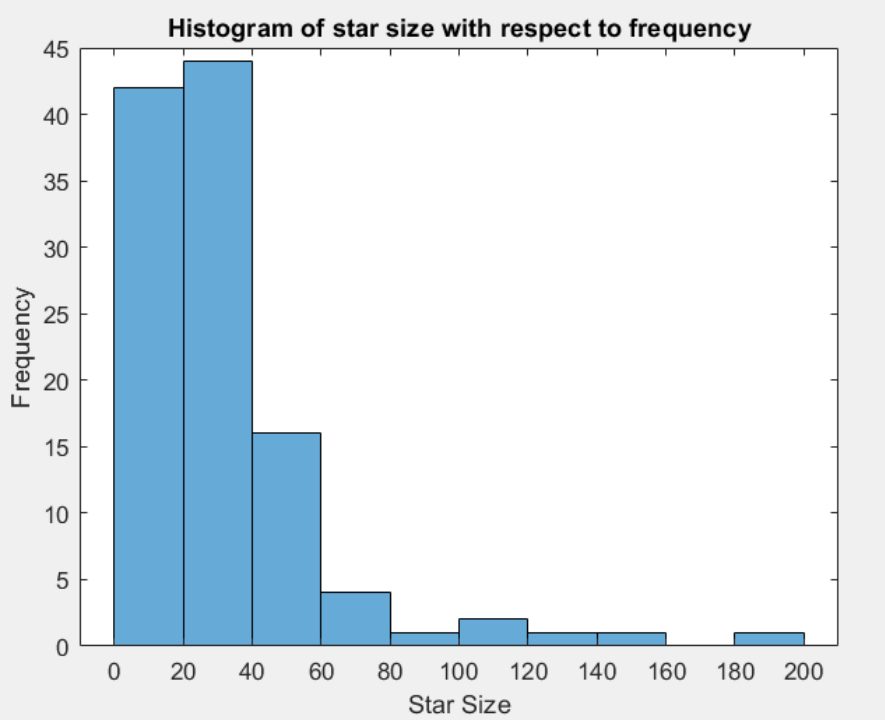


Original Image

Result

1)After passing the input to the shrinking program or the connected component analysis program. In the shrinking method all the stars reduce to a point. Therefore the no of points is the no of stars. In the case of connected component labelling each star is labelle with a different label. Thus, the unique set of labels determine the count of stars. **No of stars is 112**

2)The no of pixels that contains a specific label determines the size of each star. The no of **unique sizes is 48.**



c)PCB Analysis

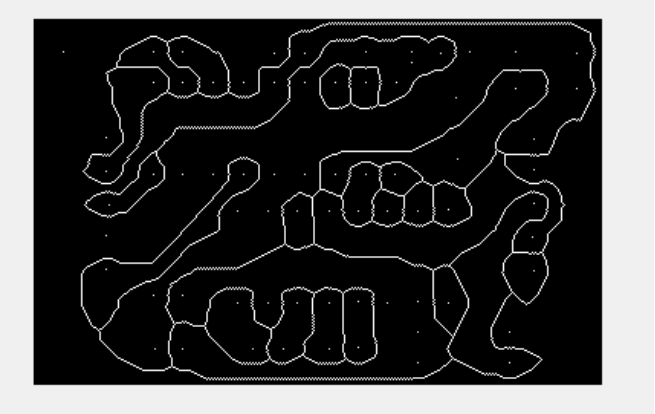
Procedure:

1)The PCB.raw image is first binarized. The shrinking algorithm is applied on the image.

2) We can see that the image after shrinking will contain dots for holes and the line segments will be connected.

3) Thus, apply connected component analysis on the resulting image to count only those components that have one pixel. **Thus, the no of holes is 72.**

4) The morphological operator to remove inner pixels is used. The no of line segments is half of the no of connected components.



Shrinking result

```
noofsegments =

     1    74

No of line segment
    37

>>
```

(d) Defect detection

The aim of the problem statement is to find a missing teeth algorithm.

Procedure

1)Binarize the image

2)Using the flood fill operation on the background pixels of the binary image, this morphological operator fills the holes (dark pixels surrounded by lighter pixels)

3)Find circles in the image by using imfindcircles. It finds the circles using circular hough transform.

Circular Hough transform algorithm is as follows: [Source: https://en.wikipedia.org/wiki/Circle_Hough_Transform]

Find circle parameter with unknown radius [edit]

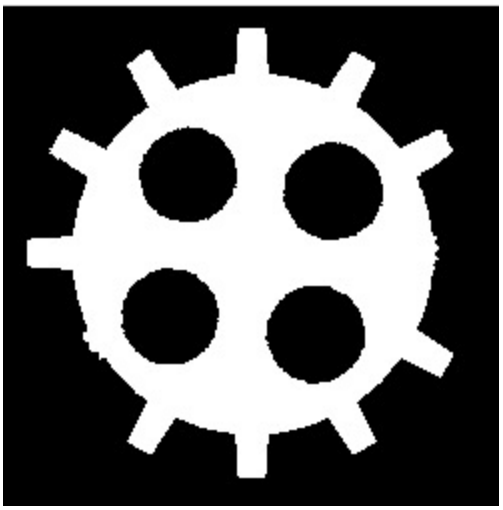
Since the parameter space is 3D, the accumulator matrix would be 3D, too. We can iterate through possible radii; for each radius, we use the previous technique. Finally, find the local maxima in the 3D accumulator matrix. Accumulator array should be A[x,y,r] in the 3D space. Voting should be for each pixels, radius and theta A[x,y,r] += 1

The algorithm :

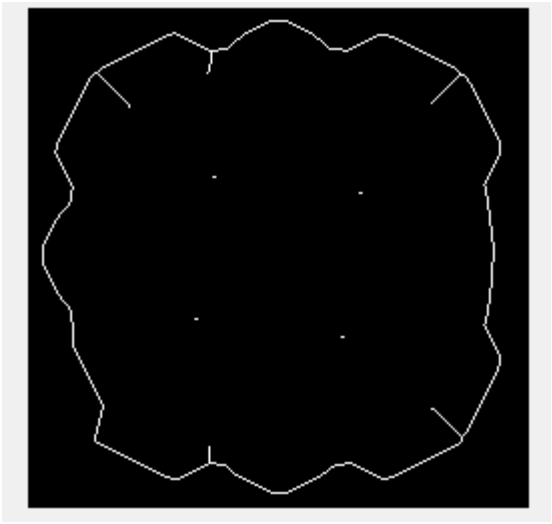
1. For each A[a,b,r] = 0;
2. Process the filtering algorithm on image Gaussian Blurring, convert the image to grayscale (grayScaling), make Canny operator, The Canny operator gives the edges on image.
3. Vote the all possible circles in accumulator.
4. The local maximum voted circles of Accumulator A gives the circle Hough space.
5. The maximum voted circle of Accumulator gives the circle.

The Voting :

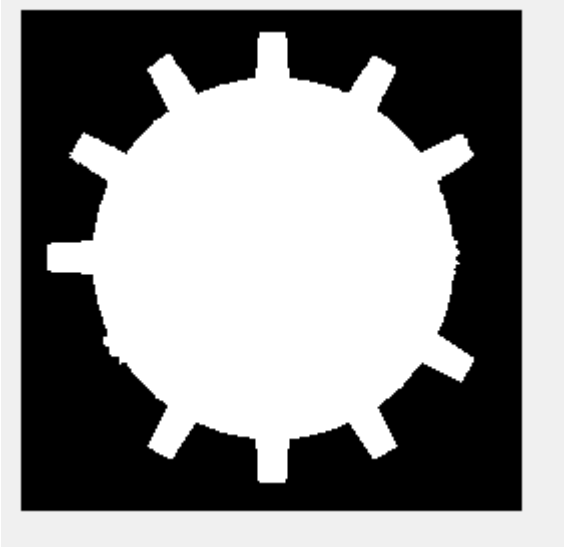
```
For each pixel(x,y)
  For each radius r = 10 to r = 60 // the possible radius
    For each theta t = 0 to 360  // the possible  theta 0 to 360
       a = x - r * cos(t * PI / 180); //polar coordinate for center
       b = y - r * sin(t * PI / 180);  //polar coordinate for center
       A[a,b,r] +=1; //voting
    end
  end
end
```



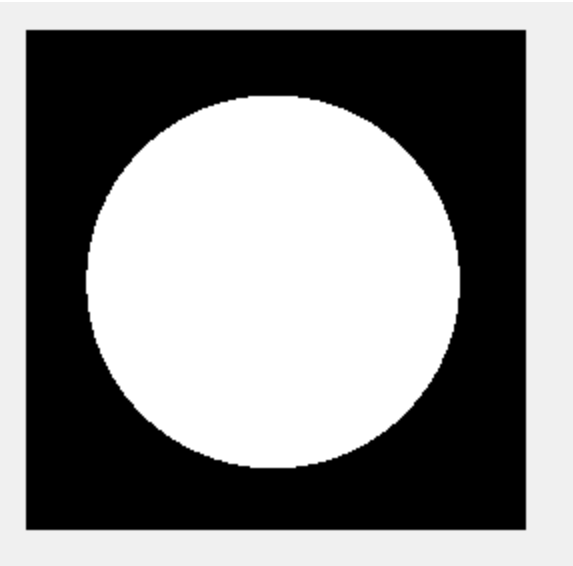
Original Image



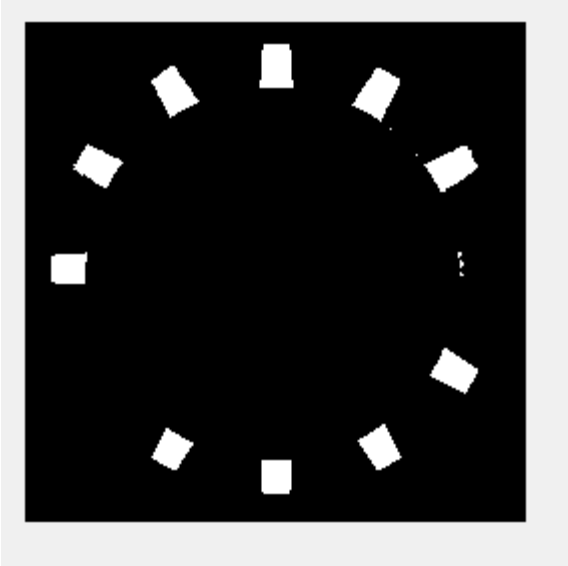
Four center points



Filled Image



Circle is detected



Positions of the gear teeth

4)A binary image of the circle is created using the radius found in step 3.

5)The circle image is subtracted from the Original Binary Image to give the geartooth.

6)Using connected component labelling, the no of large connected components are found.

The result was 10. **Thus, the number of gear tooth are 10. This is less than 12 so the gear is defective.**

Command Window

New to MATLAB? See resources for Getting Started.

```
>> hwgeartooth
    Retrieving Image Geartooth.raw ...

centers1 =

  126.4077  124.0787


radii1 =

   93.2779

Defective
No of missing teeth
     2
```