CSE585: Digital Image Processing II
Computer Project 2
Homotopic Skeletonization and Shape Analysis

Aishwarya Mallampati, Mrunmayi Ekbote, Wushuang Bai

Date:02/21/2020

---

1 Objectives

This project aims to familiarize certain concepts of digital image processing such as skeletonization, granulometry, pattern spectrum(pecstrum), shape complexity among other filters that we have learnt in class. The main objectives of the project include the following:

- To perform thinning function on an image to find its skeleton
- To implement shape analysis for two sets of images

2 Methods

2.1 Homotopic Skeletonization

2.1.1 Theory and Algorithms

The morphological skeletonization algorithm sk(X) as given in L6-1 is defined as follows:

$$sk(X) = \bigcup_{0<=n<=N} S_K(X) \tag{2.1}$$

where $S_n(X) = (X \ominus nB) - (X \ominus nB)_B$
In order to obtain this, consider thinning of an image X by B_i which is given as

$$X \circ B_i = X - X \otimes B_i \tag{2.2}$$

where $B_i = \{B_i^f, B_i^b\}$ is a set of two structuring elements and $X \otimes B_i$ is the hit-or-miss transform given by

$$X \otimes B_i = (X \ominus B_i^f) \cap (X^c \ominus B_i^b) \tag{2.3}$$

The thinning operation is performed on image X using a set of structural elements until the skeleton of the input image is obtained i.e. the image cannot be thinned anymore without loosing homotopy.

2.1.2 Matlab

In order to perform homotopic skeletonization is matlab, two input images penn256.gif and bear.gif are used as input images.
The code for this section is written in **main_skeletonization.m** file. The following steps are performed in matlab:

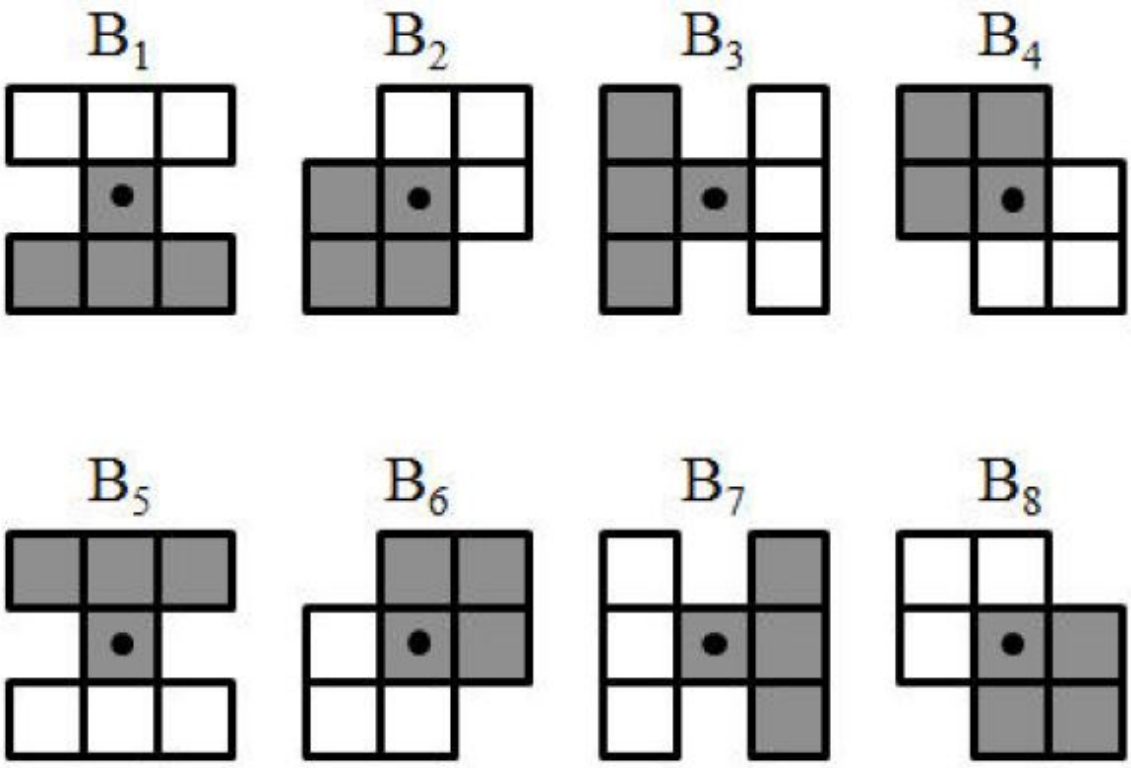

Figure 1: Structuring Elements B_i

- The two images are read from the directory.
- Structuring elements as shown in the figure 1 are created in the code by considering the shaded pixel as 1, unshaded pixel as 0 and pixels which are not considered are taken as NAN.
- To get the skeleton of an image, thinning operation(eq2.2) has to be performed on the input image over multiple iterations.
- In every iteration, hit or miss transform(eq2.3) is performed on the image and the result is used to compute the difference. $\text{main}_skeletonization.mfilecalleserosion.mfiletoerodetheir$ strips of a layer in a particular direction.Thus, each thinning iteration through the complete set of $8B_i$ isotropically "burns off" a layer around X.
- The above steps are continuously executed until no more layers can be striped off. The resultant thinned image after every iteration is compared with the resultant image in previous iteration, if both are same then the image cannot be thinned any more so the algorithm is halted. (Ex-NOR logical operation is used to detect whether the images are same or different pixel wise)
- The results are then obtained and are shown in section 3.
- **Note: Run main_skeletonization.m file to get the results**

2.2 Shape Analysis

2.2.1 Theory and Algorithm

To perform shape analysis, we need to find the size distribution, pecstrum and complexity of each object(X) in a given image. The following are the definitions of each of these terms:

- Size distribution(Granulometry) of object X:

$$U(r) = m(X_{rB}), r \in \tag{2.4}$$

  where (X_{rB}) is the opening of X by rB and $m(X_{rB})$ is the area of X_{rB}

- Pattern Spectrum(Pecstrum) of shape X:

$$f(r) = \frac{-\frac{du(r)}{dr}}{m(X)} \forall r > 0 \tag{2.5}$$

  f(r) gives the amount of area in X per component

- Shape complexity of X:

$$H(X|B) = -\sum_{i=0}^{N} f(i) log(f(i)) \tag{2.6}$$

  H(X—B) is also called as entropy.Higher the entropy - Higher the complexity of the image.

- Pattern Matching: Pecstrum can be used for pattern recognition. If $f_R(n)$ is a reference pecstrum and f(n) is the pecstrum of a new object X, then the distance d is calculated as:

$$d = \{\sum_{n=0}^{N-1} c_n (f(n) - f_R(n))^2\}^{\frac{1}{2}} \tag{2.7}$$

  The distance d can be used if the new object coincides with a reference pattern. The weights c_n can be used to emphasize certain parts of the object spectrum. The weights c_n are chosen randomly and changing them may change the results as well. The reference object that minimizes the distance is the best match for that test object.

2.2.2 Matlab

All the matlab code used for this problem is placed in Prob2-ShapeAnalysis folder. The following are the matlab files used for this problem:

- areafunction.m : used to find the area of a given image
- dilation_erosion.m : Performs dilation or erosion on the input image using the given structuring element

- fcn_sizeDistribution.m : Calculates the size distribution of given object X(implements eq2.4)
- fcn_pecstrum.m : Function used to compute the pecstrum of the given image X(implements eq2.5)
- fcn_complexity.m : Computes the shape complexity of object in a image when its pecstrum is passed to it.(implements eq2.6)
- fcn_patternRec.m : Performs pattern recognition on two given input images(implements eq2.7)

The inbuilt matlab function regionprops is used to obtain bounding boxes of objects in the image.
Note: Run problem2a_main.m and problem2b_main.m files to get results
Using regionprops inbuilt function, each object is extracted from the input images. Then area, size distribution and pecstrum of each object in the given image are computed. The pecstrum of the object is used to compute the shape complexity of each image. Pattern matching is performed between two given images using the distances and the results are recorded.
In prob2.a, shape analysis is performed on match1.gif and match3.gif images. The results are explained in section 3.1.4
In prob2.b, shape analysis is performed on shadow1.gif and shadow1rotated.gif images.The results are explained in section 3.1.5
The flowcharts for homotopic skeletonization is displayed in Figure2 and for shape analyis is displyed in Figure 3.

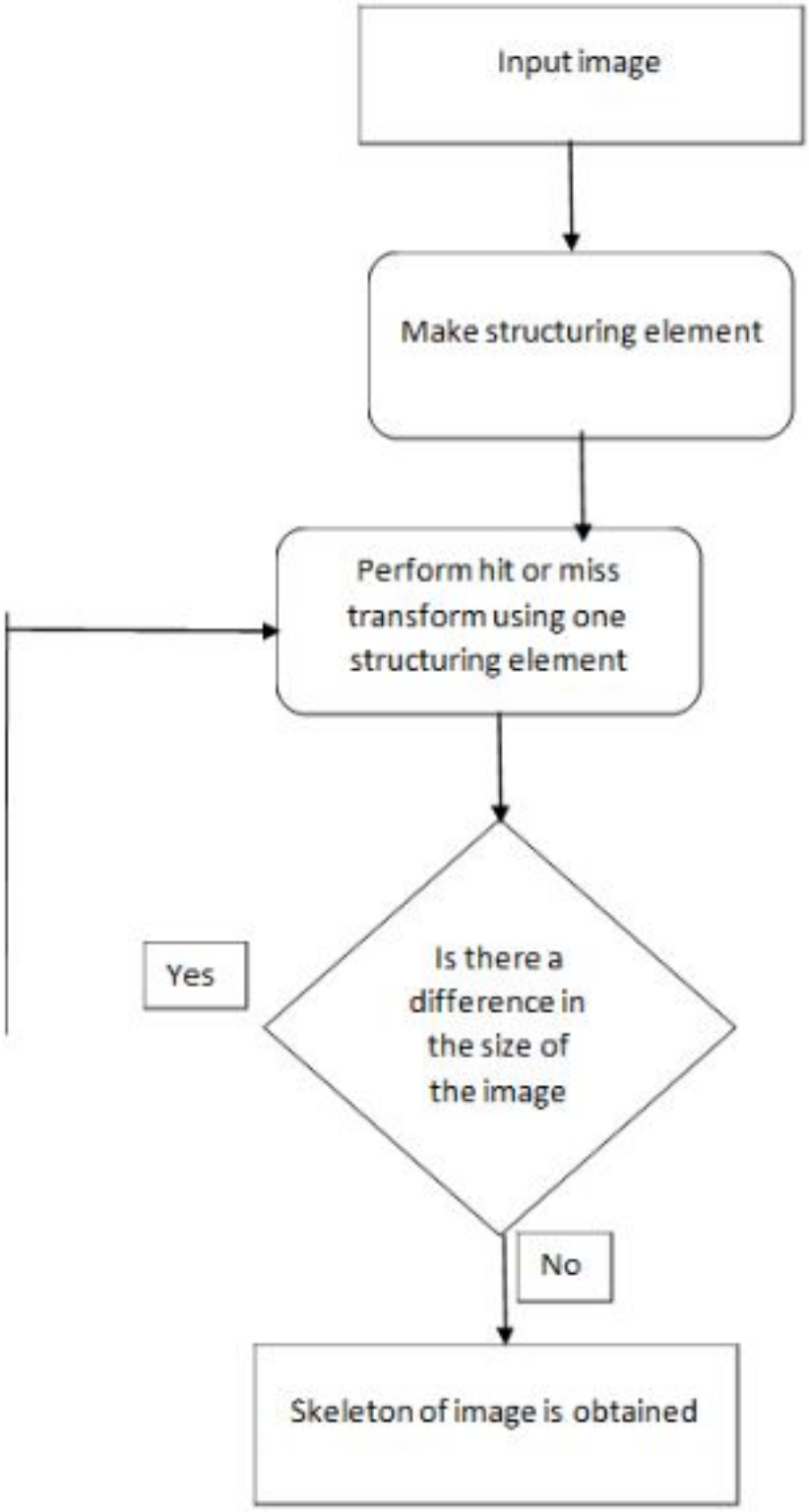

Figure 2: Flowchart : Homotopic Skeletonization

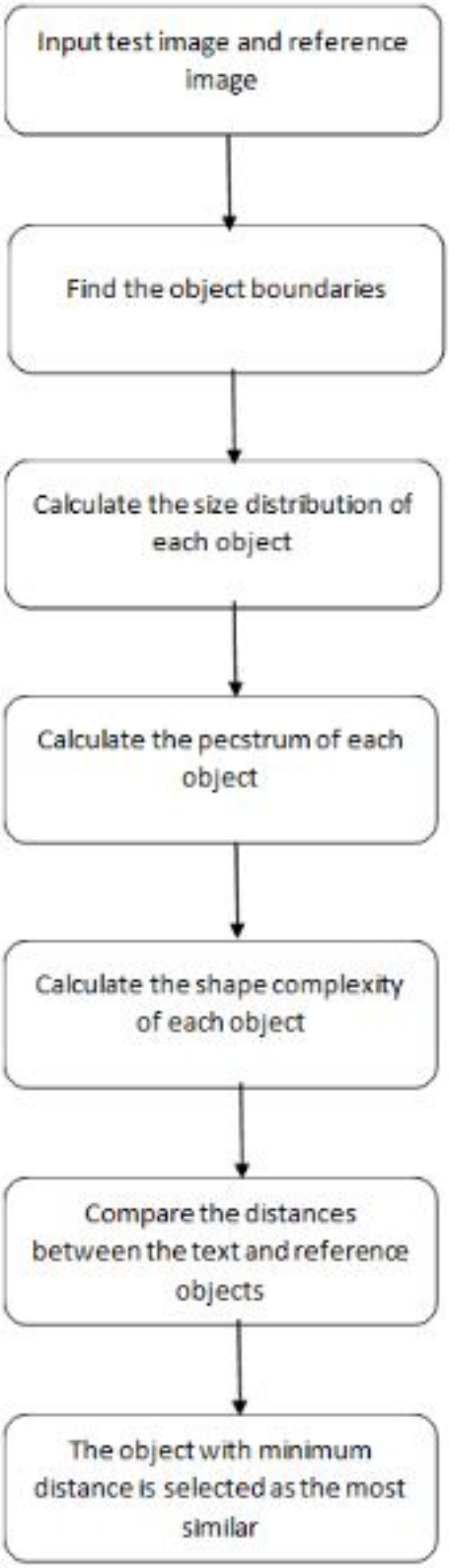

Figure 3: Flowchart : Shape Analysis

Figure 4: Input Image - penn256.gif

3 Results

3.1 Homotopic Skeletonization

To implement homotopic skeletonization, two input images penn256.gif and bear.gif are considered. These are shown in Figure 4 and Figure 5. Once user runs *main_skeletonization.m* file, the program asks the user to select the input image. Press 1 to select penn256.gif and press 2 to select bear.gif. Thinning operation is performed on the selected input image using the structuring elements shown in Figure 1. The penn256.gif image takes 8 iterations to obtain the skeleton of the image and bear.gif takes 22 iterations to obtain its skeleton. This is logically reasonable because only one layer is stripped off in each iteration and bear image is more dense than penn256 image. So, bear image takes more number of iterations and time to obtain its skeleton.
The program stores the results obtained in 2nd, 5th, 10th and final iterations in the working directory. The results are superimposed on original image and are also stored. The resultant and superimposed images obtained for penn256.gif are reported in section 3.1.1 and for bear.gif are reported in section 3.1.2

3.1.1 Penn256.gif

X_2, X_5, X_{final} for penn256.gif are displayed in Figure6, Figure8 and Figure10 respectively. Superimposition of X_2, X_5, X_{final} on the original image are displayed in Figure7, Figure9 and Figure11 respectively.
The question asks for X_{10} but our algorithm terminates at eighth iteration.

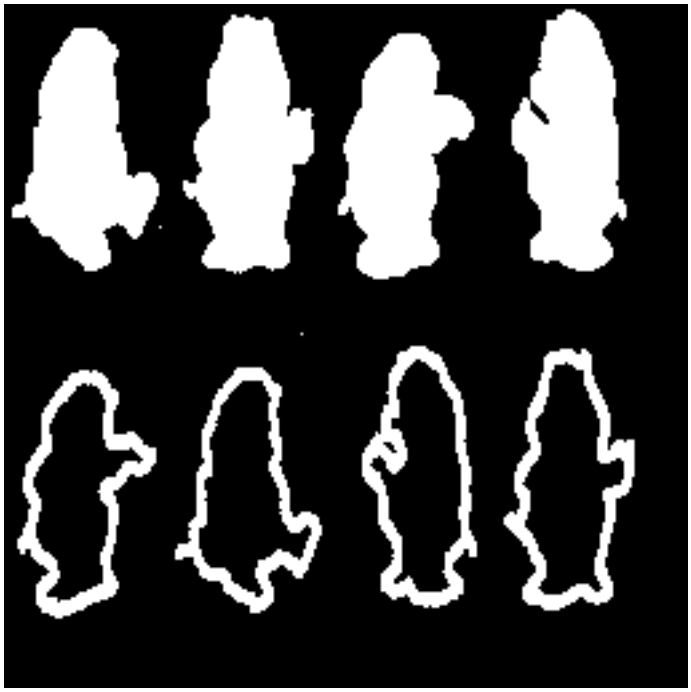

Figure 5: Input Image - bear.gif

Figure 6: X_2 of penn256 (Iteration 2)

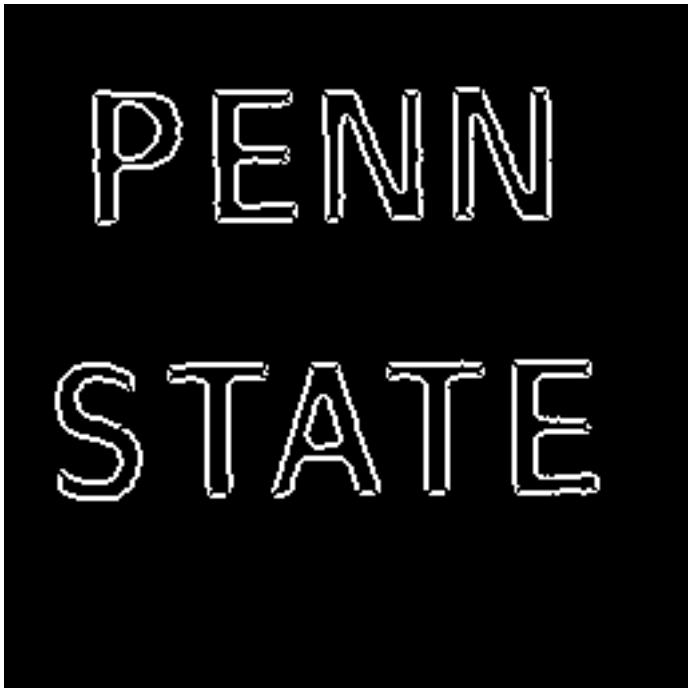

Figure 7: Superposition of X_2 on original penn256 image

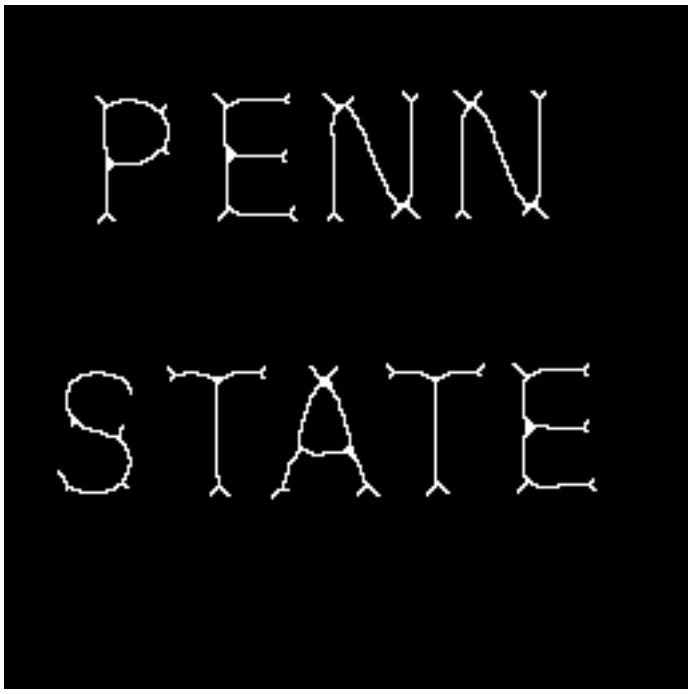

Figure 8: X_5 of penn256 (Iteration 5)

Figure 9: Superposition of X_5 on original penn256 image

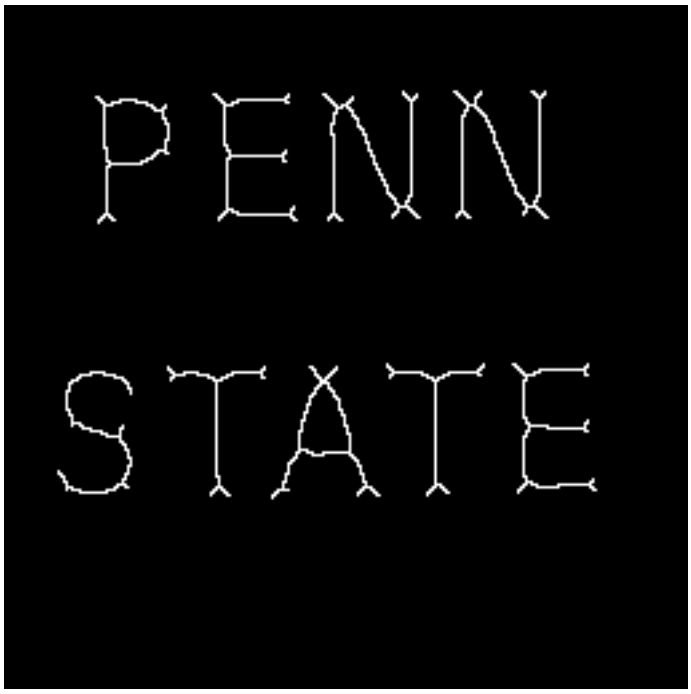

Figure 10: X_{final} of penn256 (Final skeleton)

Figure 11: Superposition of X_{final} on original penn256 image

3.1.2 bear.gif

$X_2, X_5, X_{10}, X_{final}$ for bear.gif are displayed in Figure12, Figure14, 16 and Figure18 respectively. Superimposition of $X_2, X_5, X_{10}, X_{final}$ on the original image are displayed in Figure13, Figure15, Figure17 and Figurefig. 19 respectively.

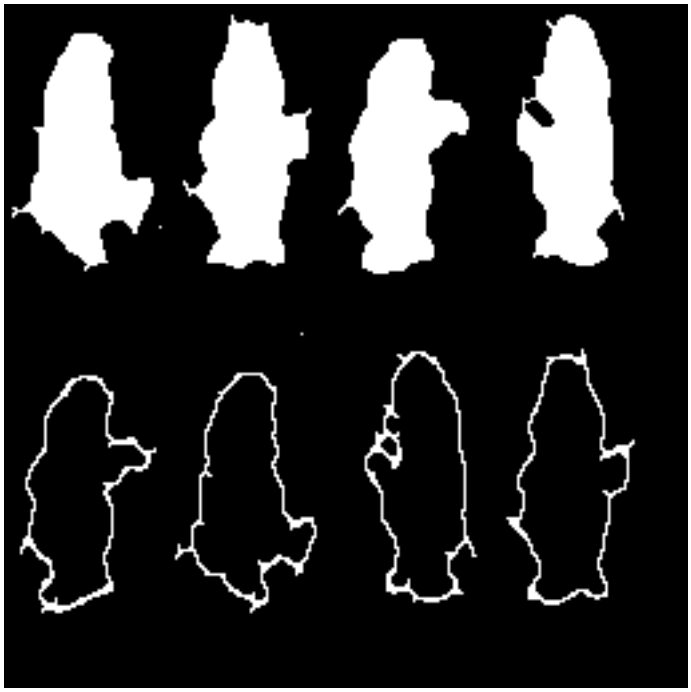

Figure 12: X_2 of bear (Iteration 2)

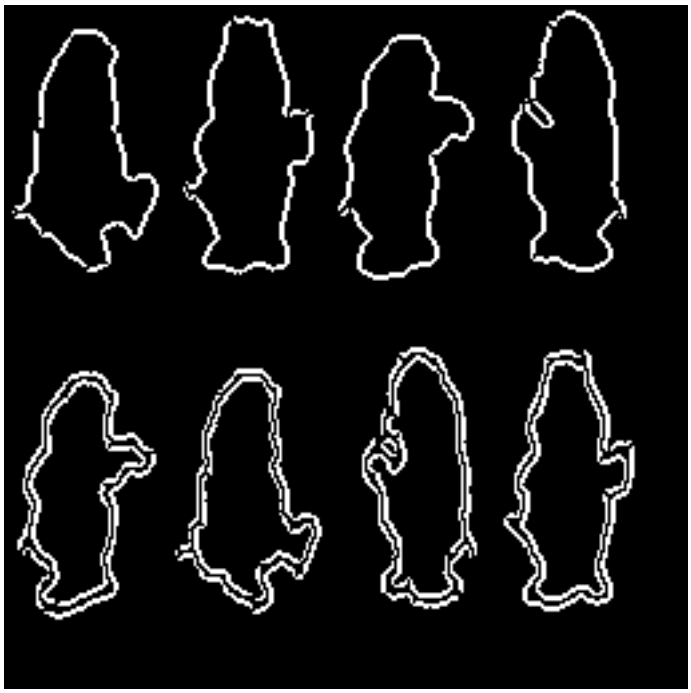

Figure 13: Superposition of X_2 on original bear image

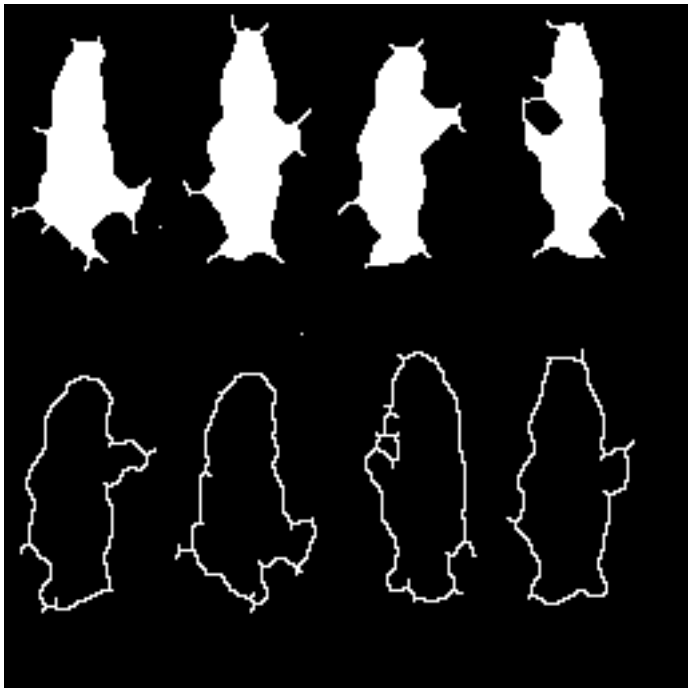

Figure 14: X_5 of bear (Iteration 5)

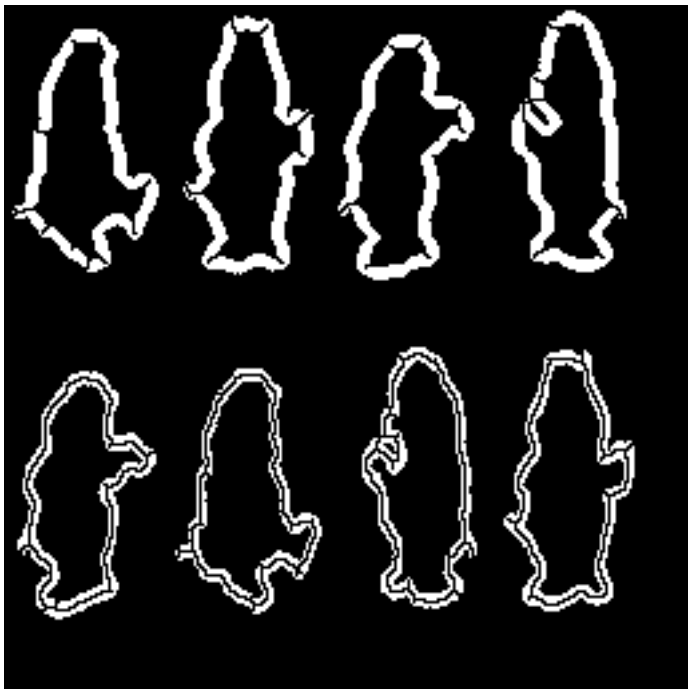

Figure 15: Superposition of X_5 on original bear image

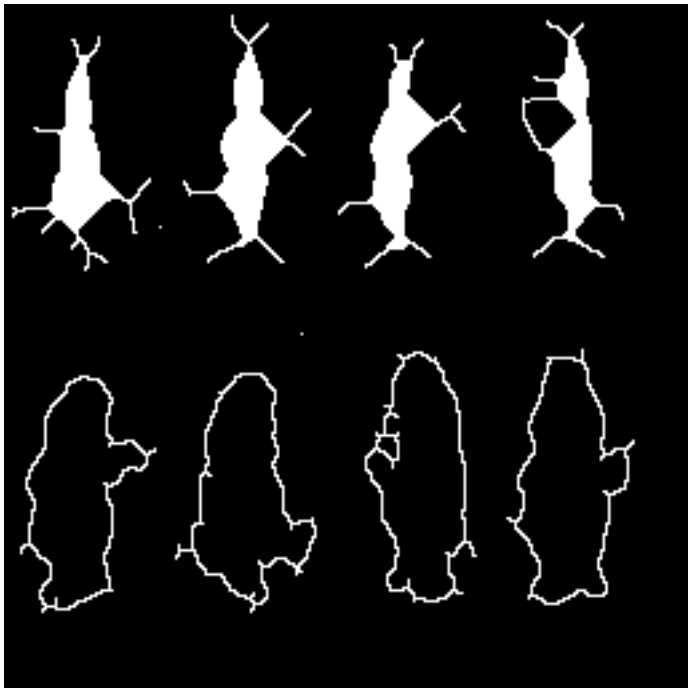

Figure 16: X_{10} of bear (Iteration 10)

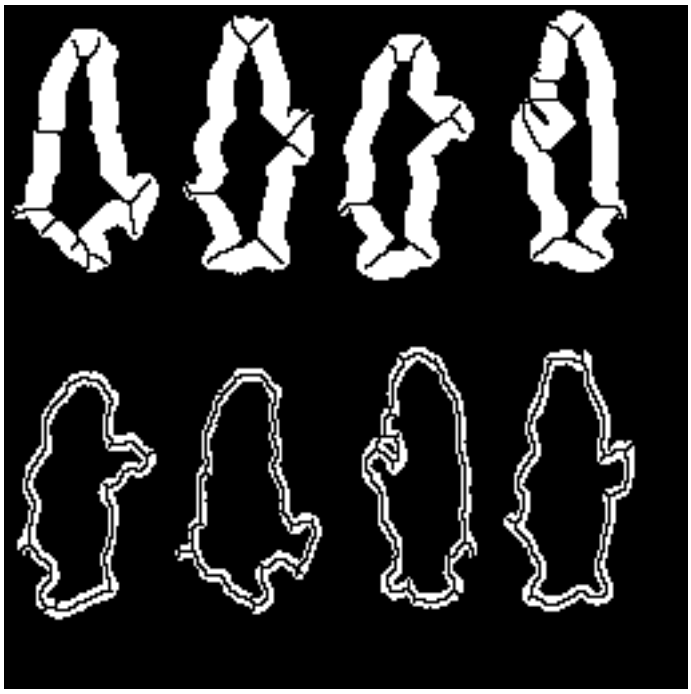

Figure 17: Superposition of X_{10} on original bear image

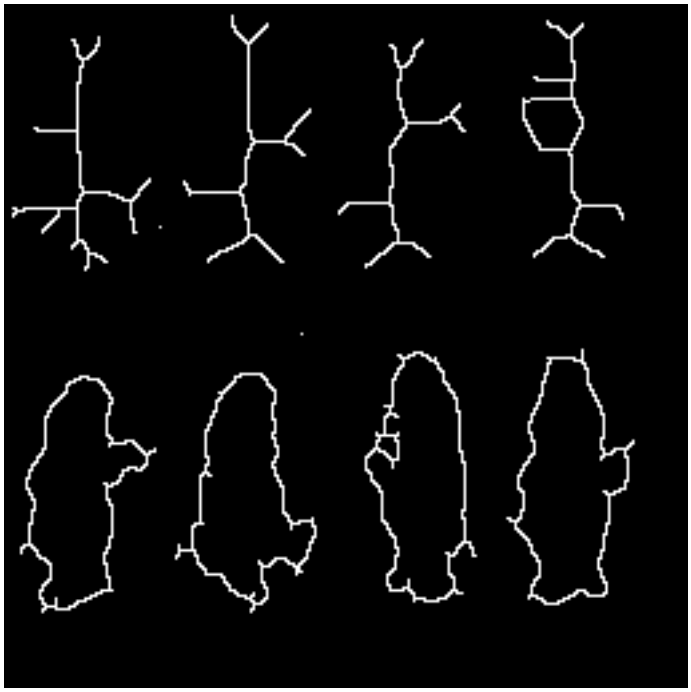

Figure 18: X_{final} of bear (Final skeleton)

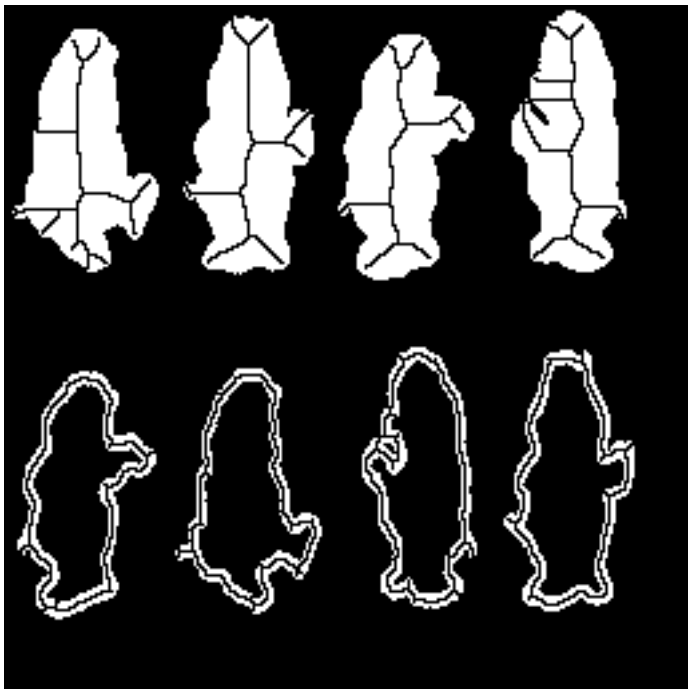

Figure 19: Superposition of X_{final} on original bear image

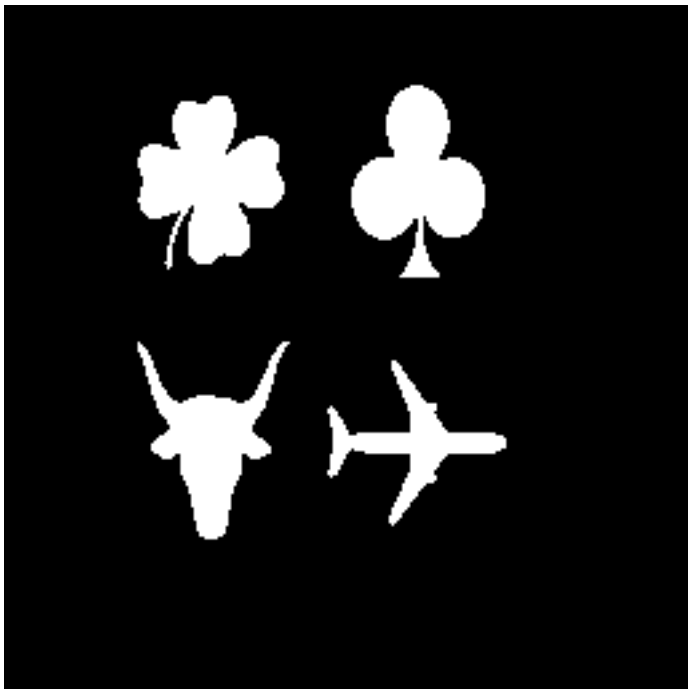

Figure 20: Input Image - match1.gif

3.1.3 Shape Analysis

3.1.4 Match1 and Match3

These results are obtained after running problem2a_main.m file. The input images are match1.gif and match3.gif which area shown in Figure20 and Figure21. Figure20 and Figure21 contain objects with same shape but in Figure21, objects are rotated and their sizes have been changed. Figure22 shows the bounding boxes around objects in match1 and Figure23 displays bounding box around objects in match3 image. The objects in these images are flower, bull, aeroplane and spade.

After extracting each object from the image, size distribution and pecstrum are computed. The size distribution and pecstrum of flower, bull, aeroplane and spade are displayed in Figure24, Figure25,Figure26 and Figure27 respectively.

The entropy of each object is also calculated in both images and displayed in table 1. From the table 1, it can be observed that the most complex object is spade as it has the highest entropy value in both the images and the least complex object is the aeroplane as it has the least entropy.

Object	Entropy in match1	Entropy in match3
Flower	1.8942	1.8336
Bull	1.869	1.8773
Aeroplane	1.5354	1.3497
Spade	1.9047	2.1672

Table 1: Shape Complexity of objects in match1 and match3

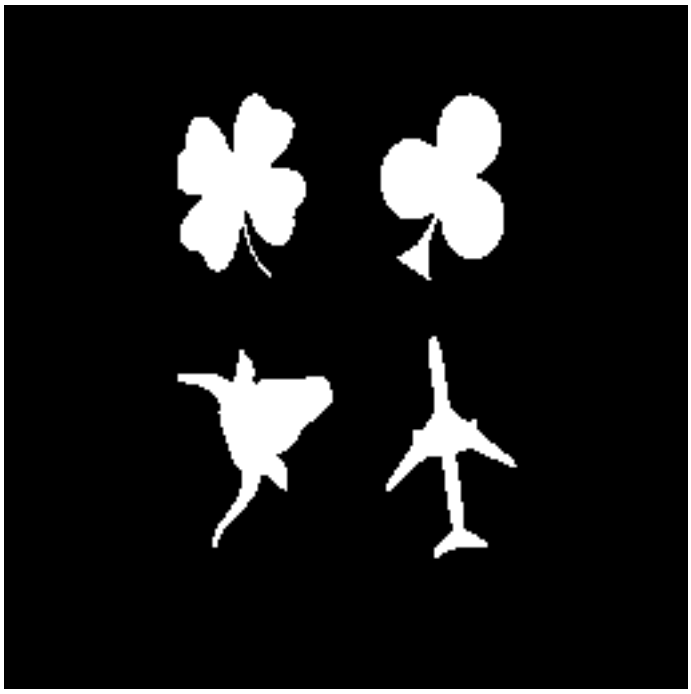

Figure 21: Input Image - match3.gif

Using artifically generated weights, pattern matching is also performed on both the images. The objects are labelled as label1=flower, bull, aeroplane, spade in match1.gif and label2 = flower, bull, spade, aeroplane in match3.gif. So, the pattern matching results are displayed in Figure28 which is screen shot taken from the console window. Three objects match correctly whereas there is a mismatch with respect to one object. Selecting a different set of weights might result in correct match for all the objects in both images.

for match1.jpg for match1.bb

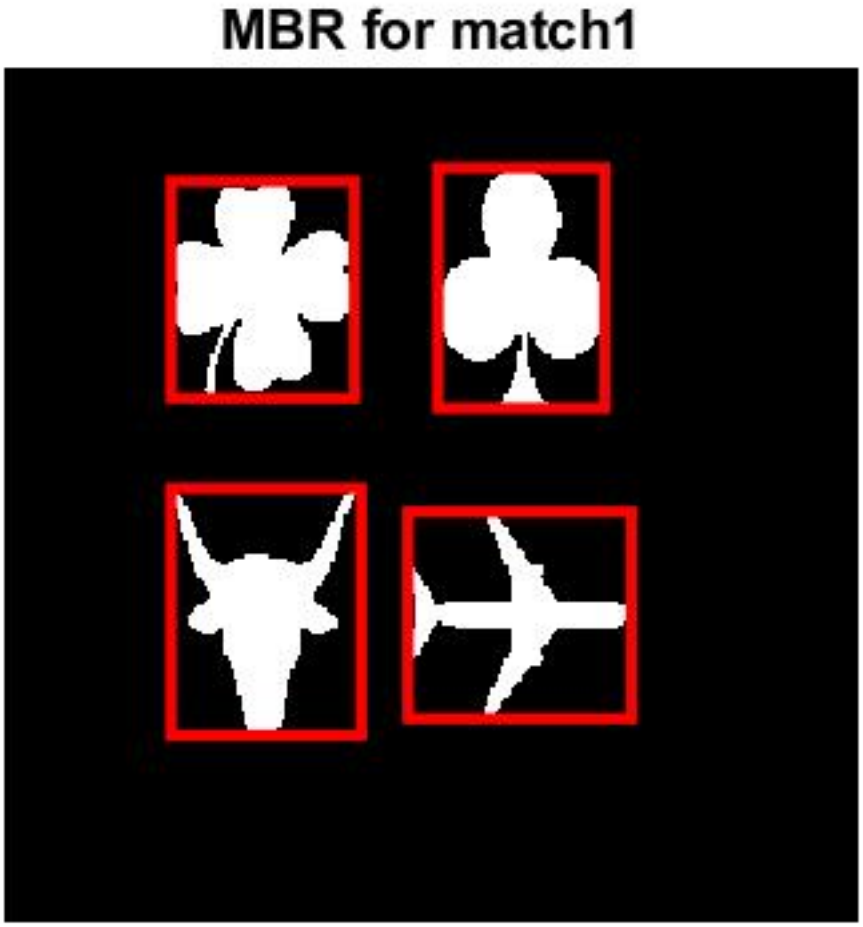

Figure 22: Minimum bounding box around objects in match1.gif

for match3.jpg for match3.bb

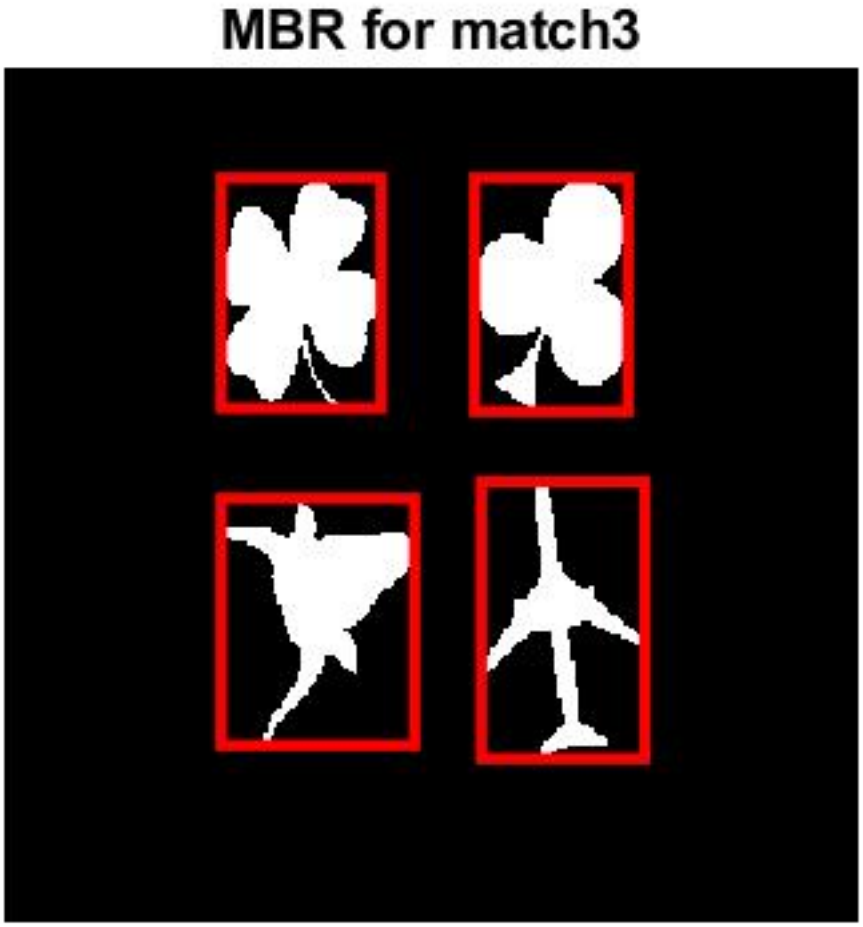

Figure 23: Minimum bounding box around objects in match3.gif

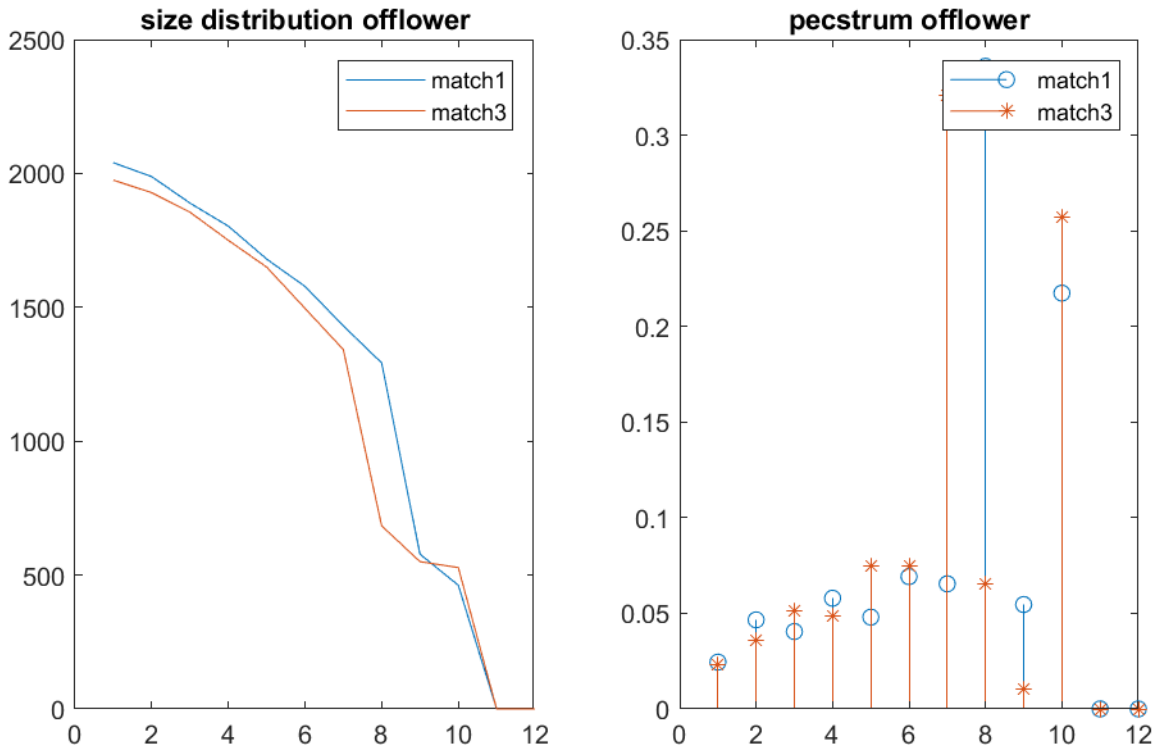

Figure 24: Size distribution and pecstrum of flower

3.1.5 Shadow1 and Shadow1rotated

Same set of operations performed in section 3.1.4 are performed on shadow1.gif and shadow1rotated.gif which are displayed in Figure 29 and Figure 30 respectively. The minimum bounding boxes around the objects in shadow1.gif and shadow1rotated.gif are displayed in Figure 31 and Figure 32 respectively.

Pecstrum and size distribution of each solid object in each of the images is computed and displayed in figures from Fig 33 to Fig 36.

The entropy of each object is computed in displayed in table 2. The objects having minimum and maximum shape complexity can be understood from the table 2 as high entropy implies highest complexity and low entropy implies least complexity.

Object	Entropy in shadow1	Entropy in shadow1rotated
1	1.9575	1.7134
2	2.0954	2.1391
3	1.9048	2.1012
4	2.1032	1.9381

Table 2: Shape Complexity of solid objects in shadow1 and shadow1rotated

Using artifically generated weights, pattern matching is also performed on both the images.So, the pattern matching results are displayed in Figure37 which is screen shot

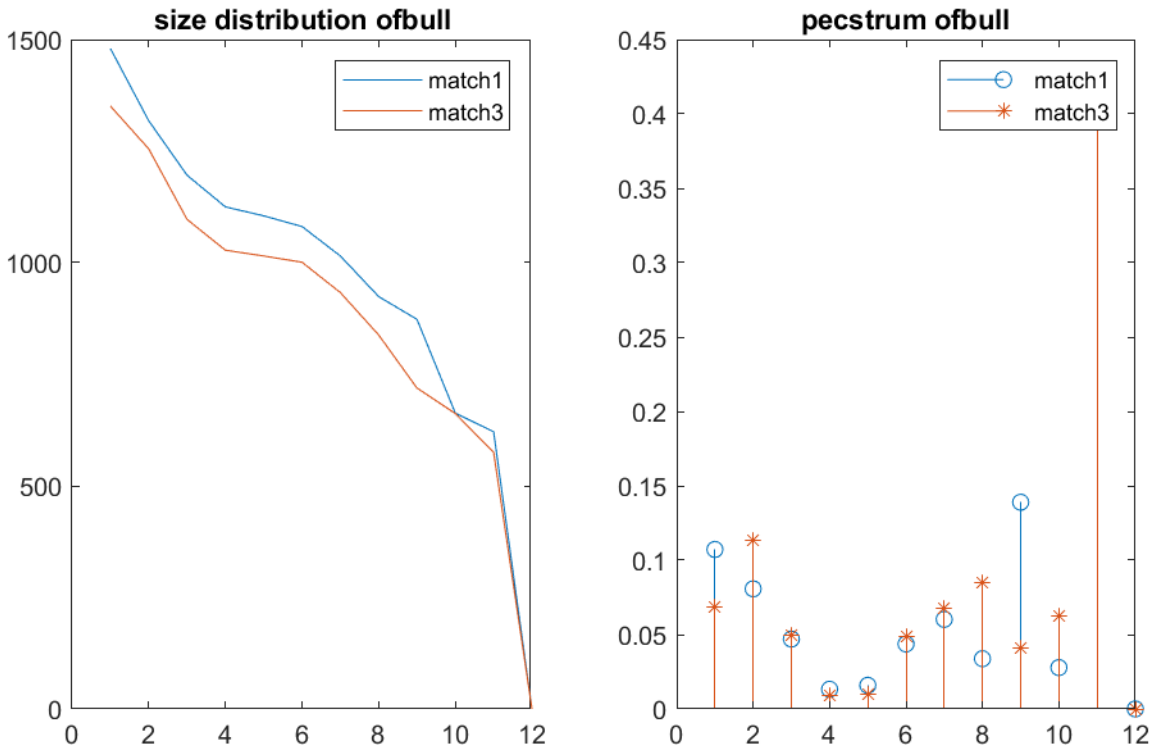

Figure 25: Size distribution and pecstrum of bull

taken from the console window. Three objects match correctly whereas there is a mismatch with respect to one object. Selecting a different set of weights might result in correct match for all the objects in both images.

4 Conclusion

After completing the necessary tasks to meet the objectives of this project, the following conclusions can be made:

- Different skeletons can be produced of the same object using different structuring elements.
- Skeletons can be used to identify several structures which can be useful during performing pattern recognition and classification.
- Shape analysis is an effective technique to match similar objects. Moreover, while computing the distance between test objects and reference objects, weights can be used to give importance to a particular component in the image. Therefore, better results can be obtained.

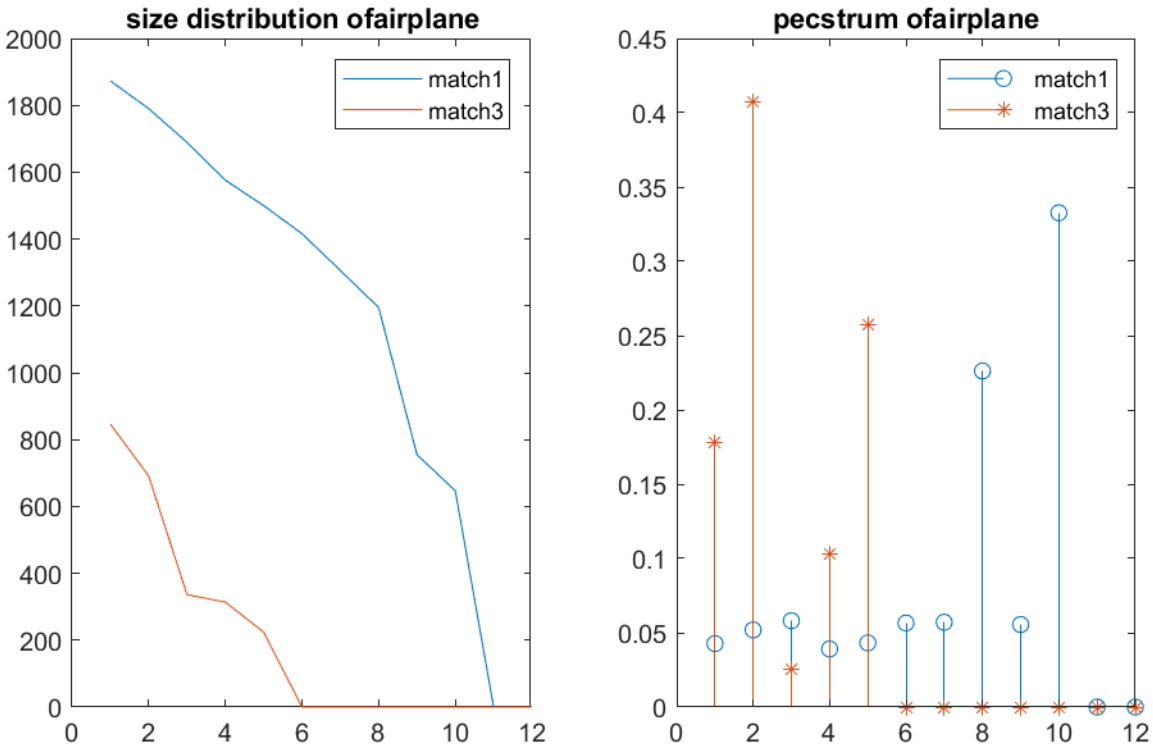

Figure 26: Size distribution and pecstrum of aeroplane

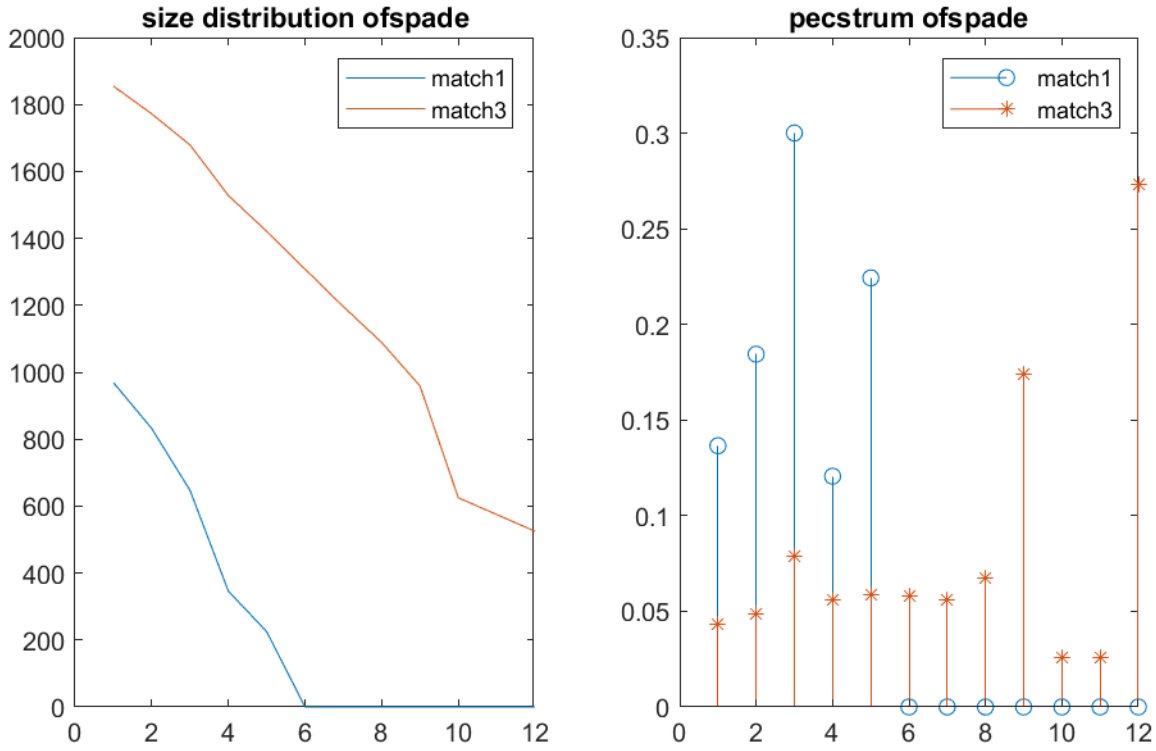

Figure 27: Size distribution and pecstrum of spade

```
-- -- -- -- -- -- -- -- -- -- -- -- -- --
Object Number 1 in match1.gif is matching Object Number 1 in match3.gif
Object Number 2 in match1.gif is matching Object Number 2 in match3.gif
Object Number 3 in match1.gif is matching Object Number 4 in match3.gif
Object Number 4 in match1.gif is matching Object Number 1 in match3.gif
>>
```

Figure 28: Pattern matching results of match1 with match3

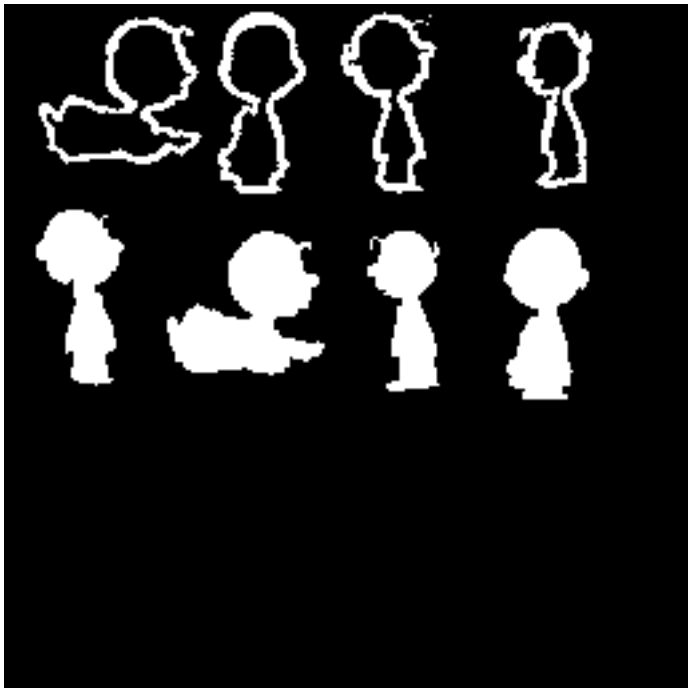

Figure 29: Input Image - shadow1.gif

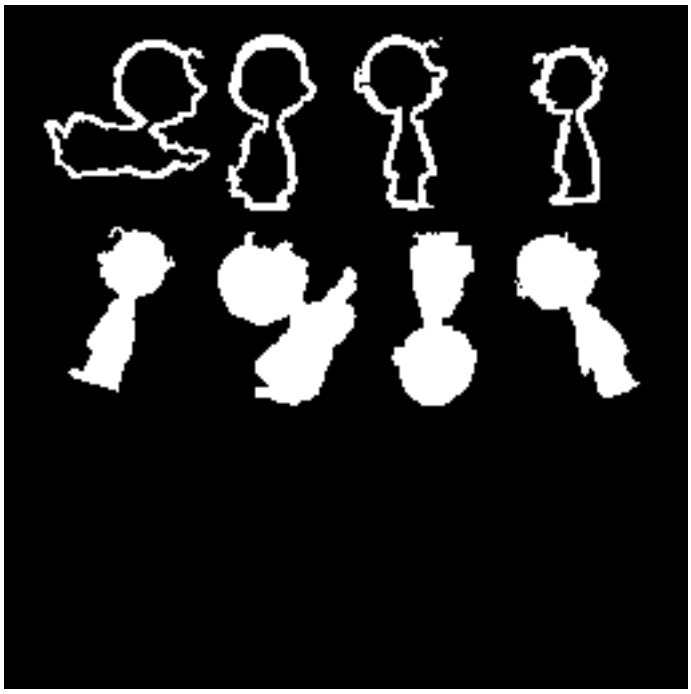

Figure 30: Input Image - shadow1rotated.gif

for shadow1.jpg for shadow1.bb

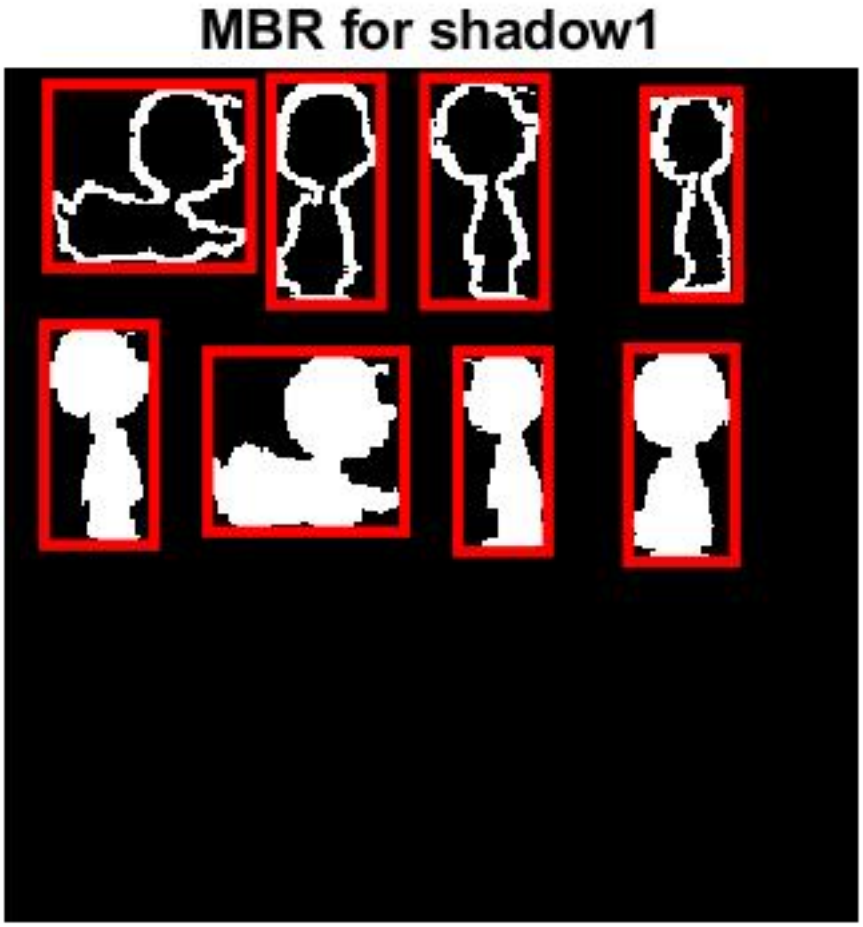

Figure 31: Minimum bounding box around objects in shadow1.gif

for shadow2.jpg for shadow2.bb

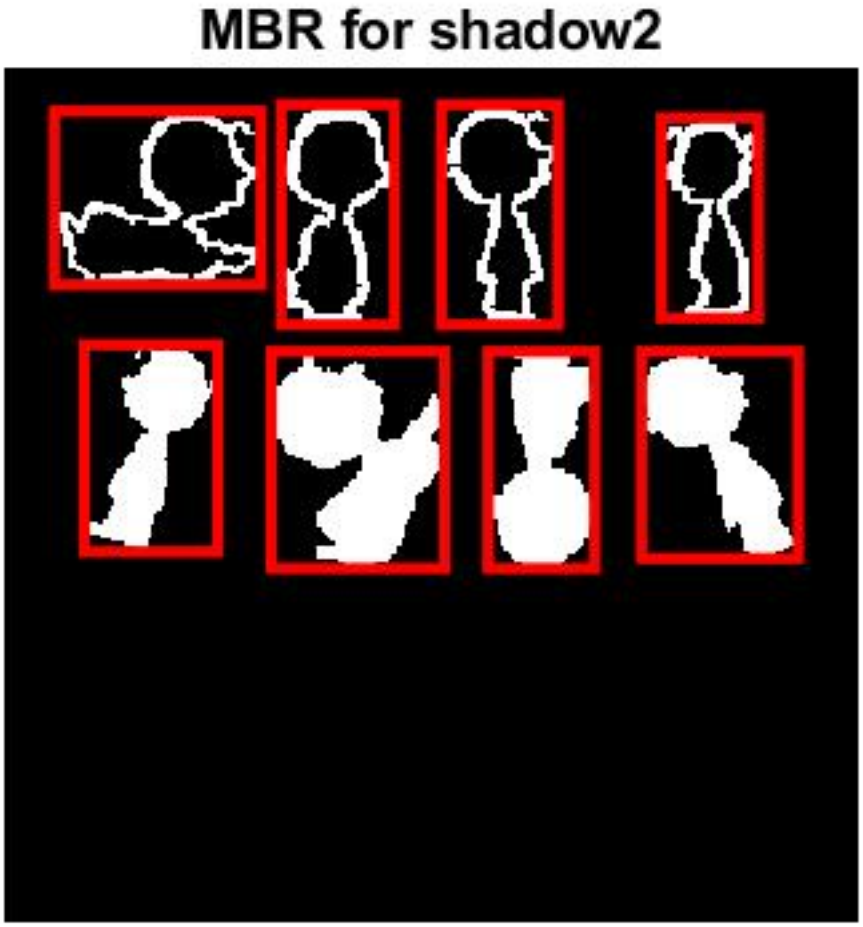

Figure 32: Minimum bounding box around objects in shadow1rotated.gif

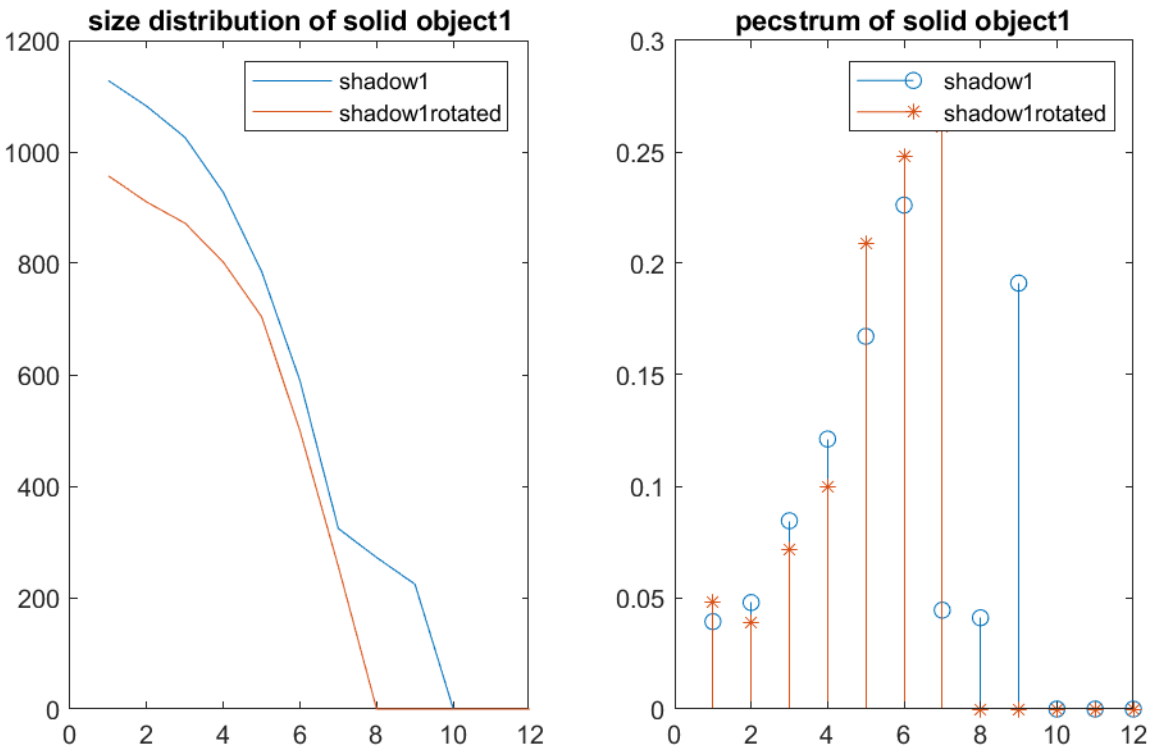

Figure 33: Size distribution and pecstrum of solid object 1

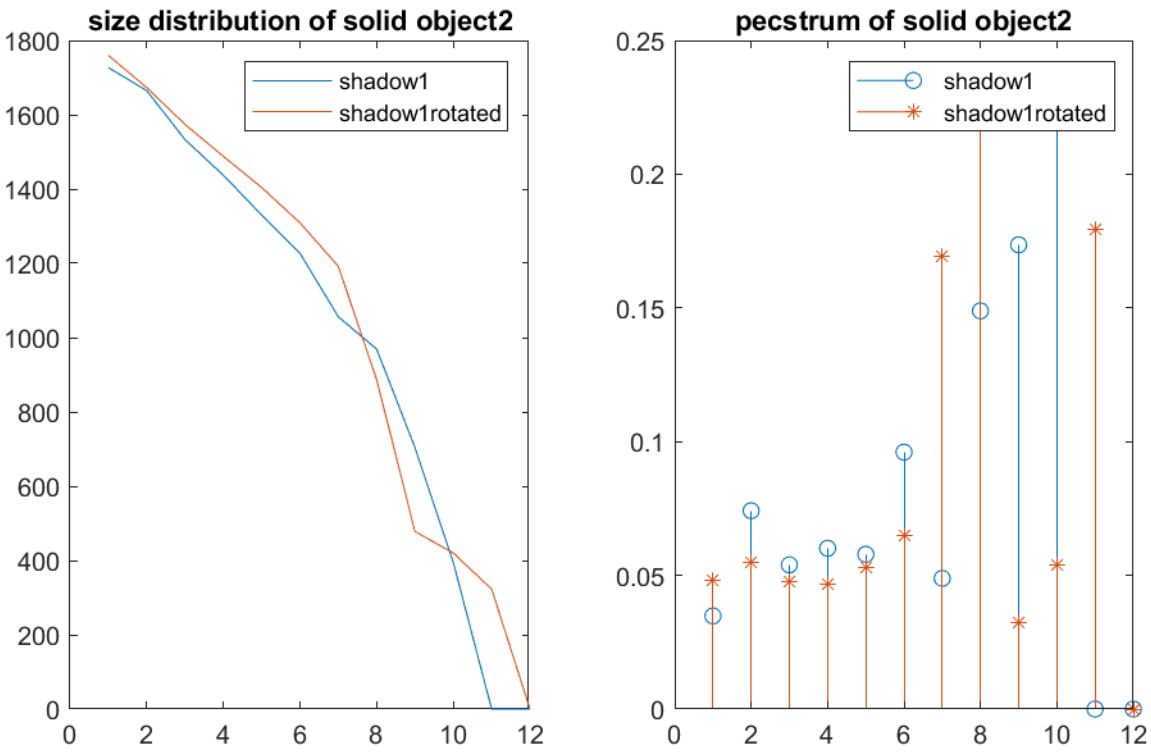

Figure 34: Size distribution and pecstrum of solid object 2

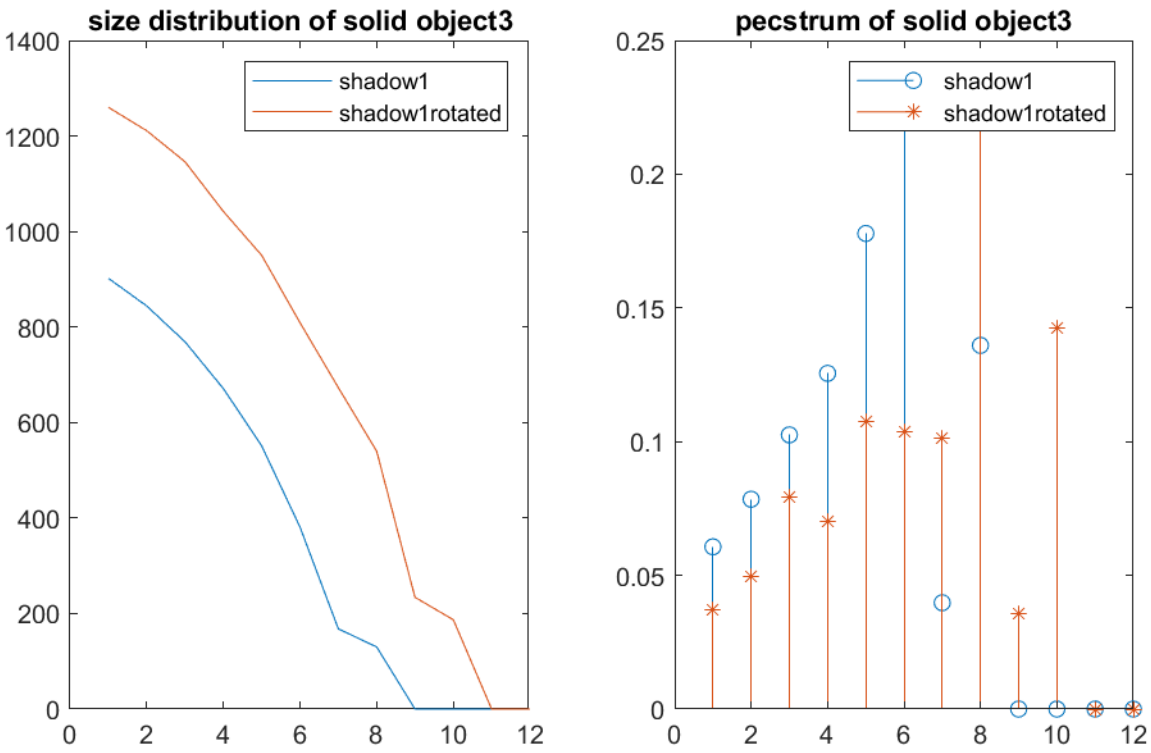

Figure 35: Size distribution and pecstrum of solid object 3

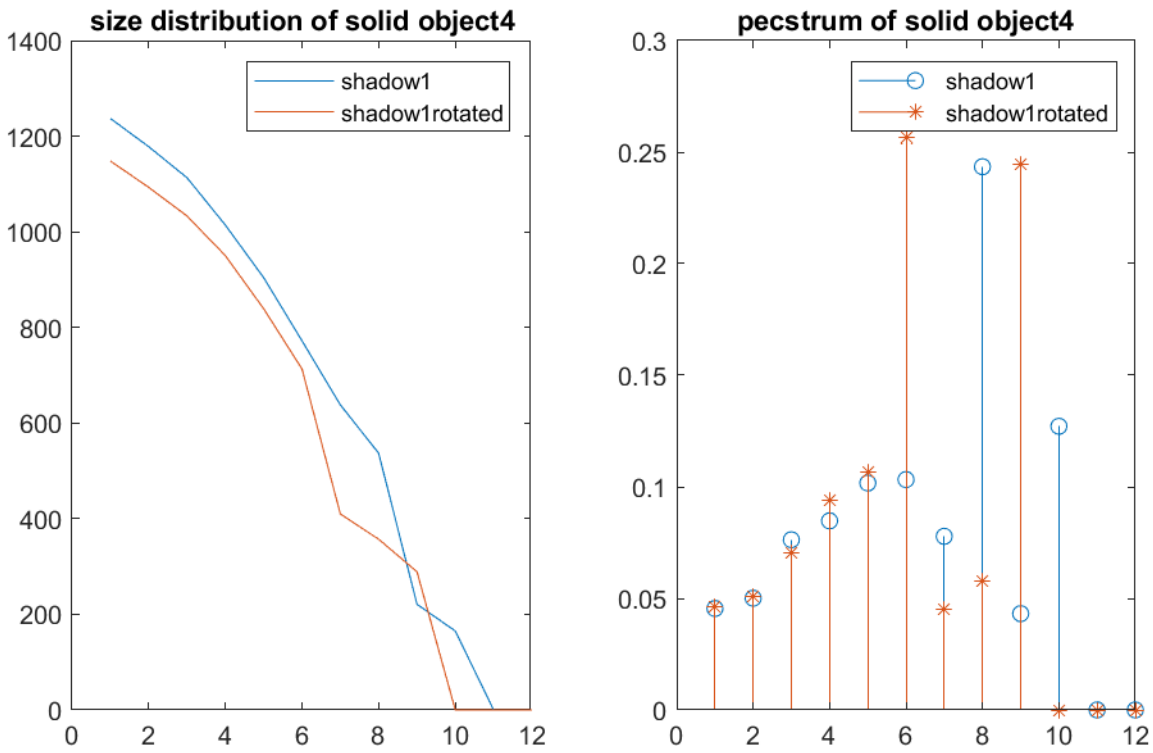

Figure 36: Size distribution and pecstrum of solid object 4

```
Solid Object Number 1 in shadow1.gif is matching Solid Object Number 1 in shadow1rotated.gif
Solid Object Number 2 in shadow1.gif is matching Solid Object Number 2 in shadow1rotated.gif
Solid Object Number 3 in shadow1.gif is matching Solid Object Number 1 in shadow1rotated.gif
Solid Object Number 4 in shadow1.gif is matching Solid Object Number 4 in shadow1rotated.gif
```

Figure 37: Pattern matching results of shadow1 with shadow1rotated