Morphology

Types of Morphological Operations

- *Morphology* is a broad set of image processing operations that process images based on shapes.
- Morphological operations apply a structuring element to an input image, creating an output image of the same size.
- In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors.

Morphological Dilation and Erosion

- The most basic morphological operations are *dilation* and *erosion*.
- Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries.
- The number of pixels added or removed from the objects in an image depends on the size and shape of the *structuring element* used to process the image.

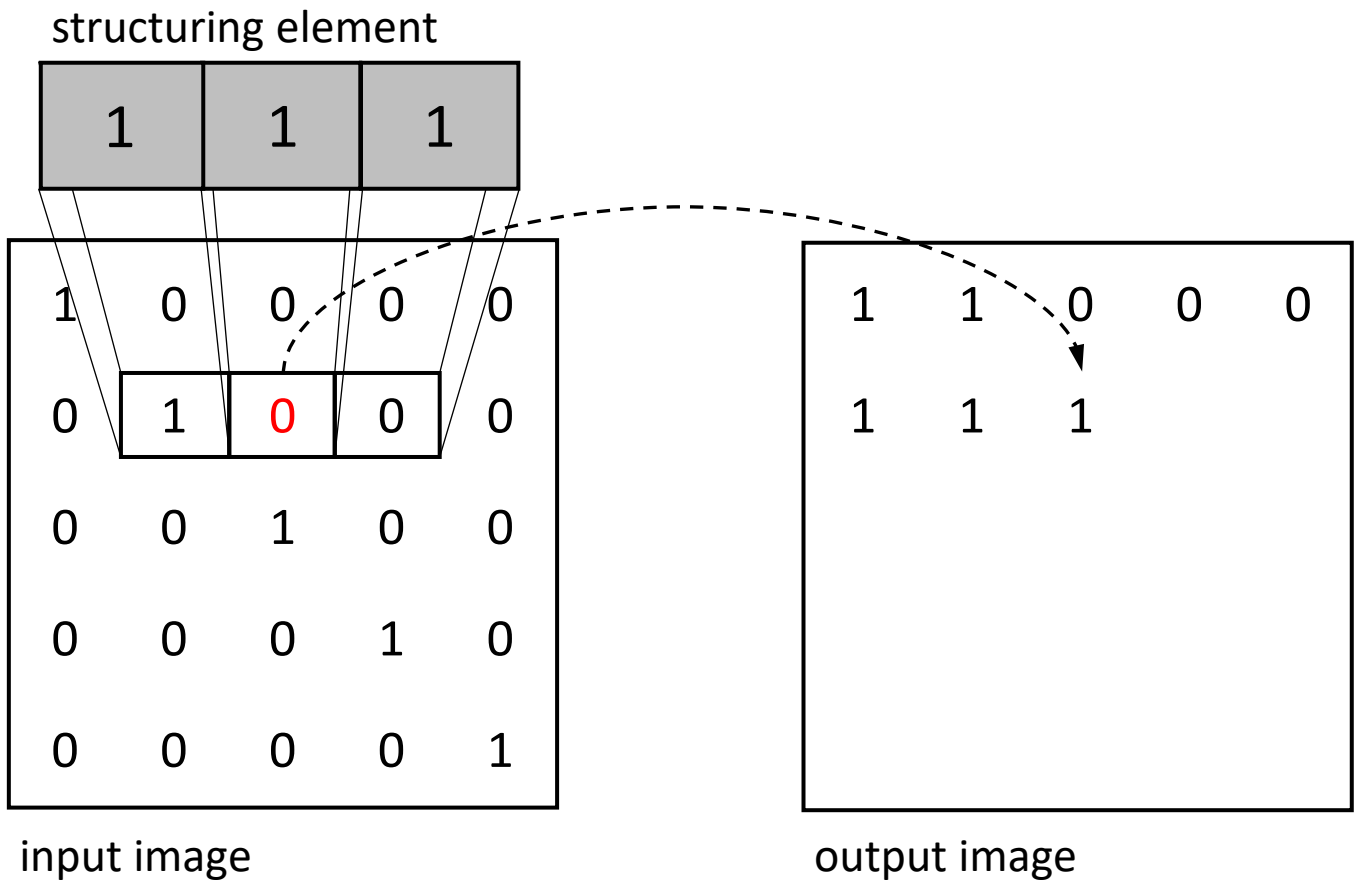
Morphological Dilation and Erosion

- In the morphological dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image.
- The rule used to process the pixels defines the operation as a dilation or an erosion.

Dilation

- The value of the output pixel is the *maximum* value of all pixels in the neighborhood.
- In a binary image, a pixel is set to 1 if any of the neighboring pixels have the value 1.
- Morphological dilation makes objects more visible and fills in small holes in objects.

Binary Dilation

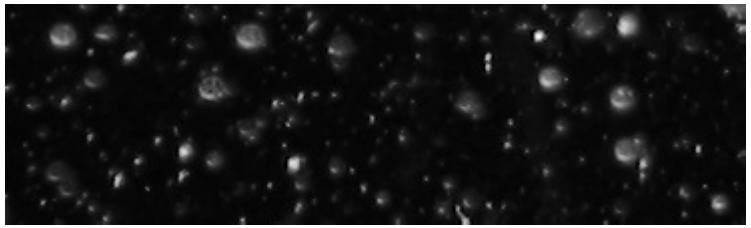


Binary Dilation

- The structuring element defines the neighborhood of the pixel of interest, (shown in red).
- The dilation function applies the appropriate rule to the pixels in the neighborhood and assigns a value to the corresponding pixel in the output image.
- The morphological dilation function sets the value of the output pixel to 1 because the maximum of the pixels in the neighborhood defined by the structuring element is 1.

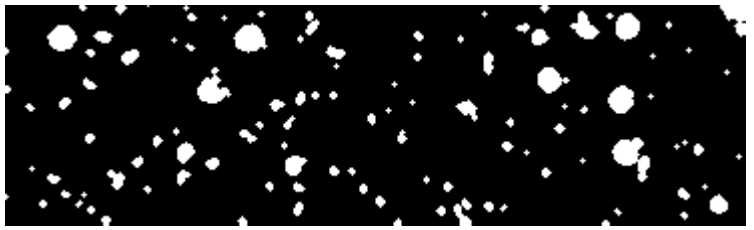
Binary Dilation Example

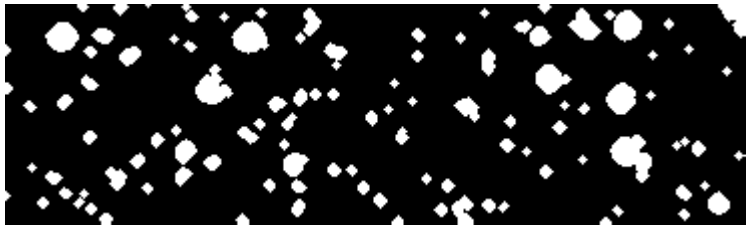
grayscale image

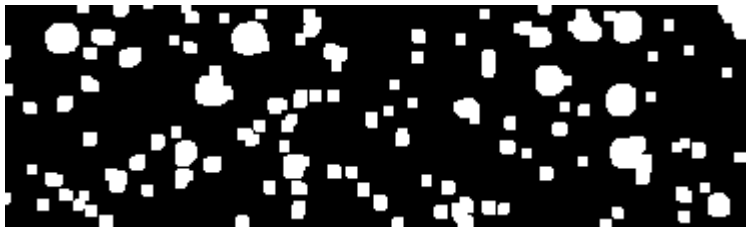


binary image

dilated image







structuring element

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

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

$$\begin{matrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{matrix}$$

Exercise 1

1. Read in the snowflakes image:

```
I = imread('snowflakes.png');
```

2. Convert it to a binary image:

```
BW = I>64;
```

3. Create a disk structuring element with radius 3:

```
se = strel('disk',3);
```

4. Dilate the binary image using the structuring element:

```
BW_d = imdilate(BW,se);
```

5. Display the dilated image:

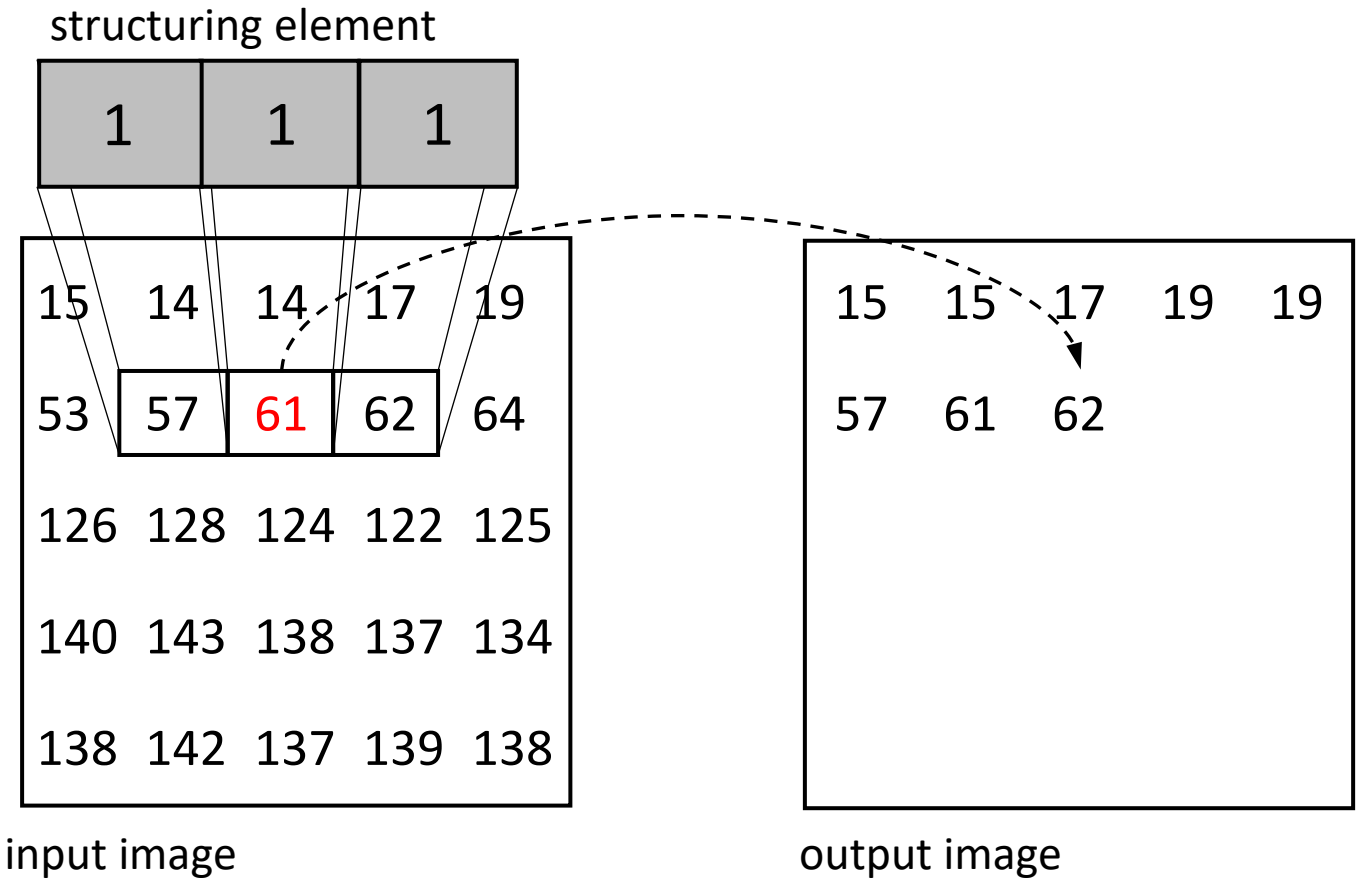
```
imshow(BW_d)
```

6. repeat steps 3, 4 and 5 using:
   - a disk structuring element with radius 2
   - a 7x7 square structuring element
   - a line structuring element with length 5 and angle 45 degrees

Grayscale Dilation

- The function applies the rule to the input pixel's neighborhood and uses the maximum value of all the pixels in the neighborhood as the value of the corresponding pixel in the output image.
- Grayscale dilation results in an increased overall brightness, expanded bright regions, and loss of small dark details.

Grayscale Dilation



Grayscale Dilation Example

grayscale image

structuring element

0	1	0
1	1	1
0	1	0

0	0	1	0	0
0	1	1	1	0
1	1	1	1	1
0	1	1	1	0
0	0	1	0	0

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

dilated image

Exercise 2

1. Read in the retina image:

```
RGB = imread('retina.jpg');
```

2. Convert it to a grayscale image:

```
I = rgb2gray(RGB);
```

3. Create a disk structuring element with radius 1:

```
se = strel('disk',1);
```

4. Dilate the grayscale image using the structuring element:

```
I_d = imdilate(I,se);
```

5. Display the dilated image:

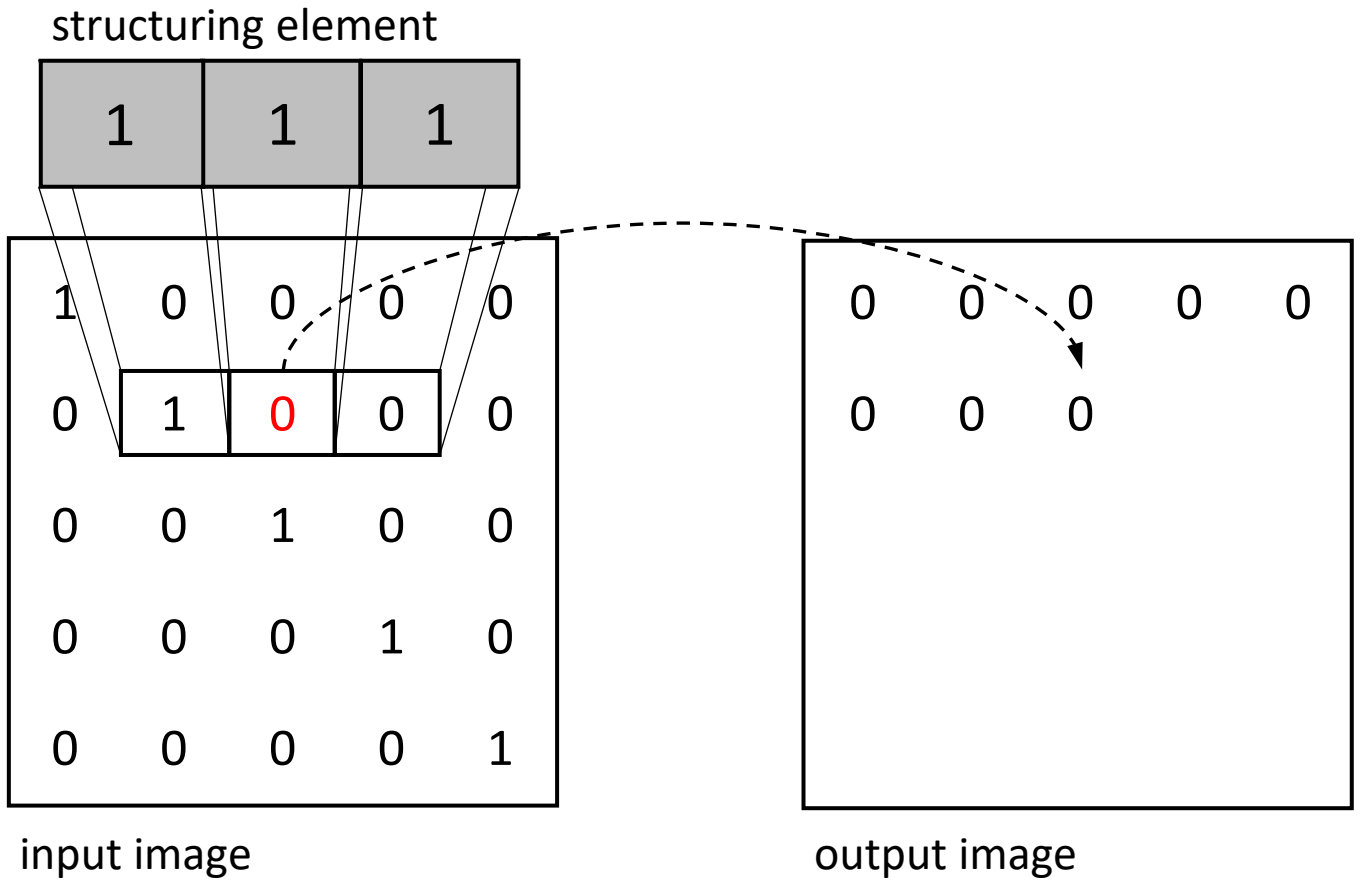
```
imshow(I_d)
```

6. repeat steps 3, 4 and 5 using:
   - a disk structuring element with radius 2
   - a disk structuring element with radius 4
   - a disk structuring element with radius 8

Erosion

- The value of the output pixel is the *minimum* value of all pixels in the neighborhood.
- In a binary image, a pixel is set to 0 if any of the neighboring pixels have the value 0.

Binary Erosion

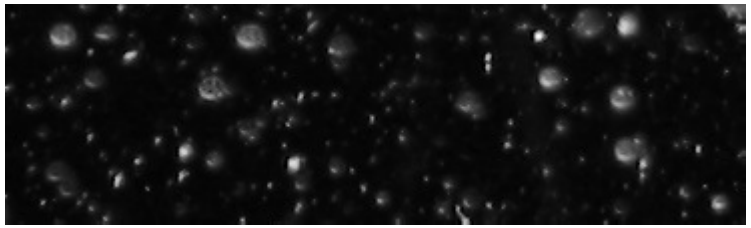


Binary Erosion

- The structuring element defines the neighborhood of the pixel of interest, (shown in red).
- The erosion function applies the appropriate rule to the pixels in the neighborhood and assigns a value to the corresponding pixel in the output image.
- The morphological erosion function sets the value of the output pixel to 0 because the minimum of the pixels in the neighborhood defined by the structuring element is 0.

Binary Erosion Example

grayscale image



binary image

eroded image

structuring element

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

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

$$\begin{matrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{matrix}$$

Exercise 3

1. Read in the snowflakes image:

```
I = imread('snowflakes.png');
```

2. Convert it to a binary image:

```
BW = I>64;
```

3. Create a disk structuring element with radius 3:

```
se = strel('disk',3);
```

4. erode the binary image using the structuring element:

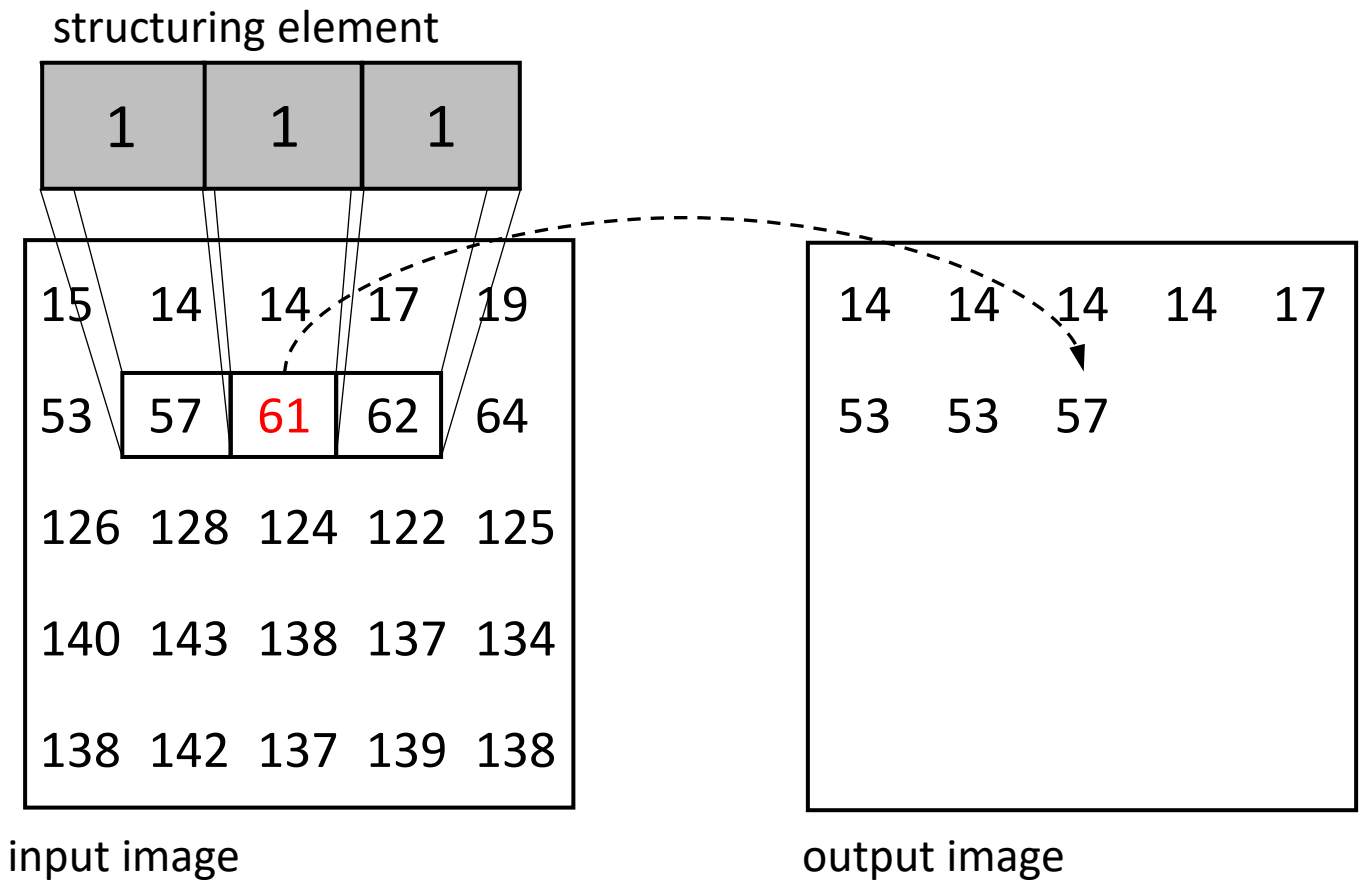
```
BW_e = imerode(BW,se);
```

5. Display the eroded image:

```
imshow(BW_e)
```

6. repeat steps 3, 4 and 5 using:
   - a disk structuring element with radius 2
   - a 7x7 square structuring element
   - a line structuring element with length 5 and angle 45 degrees

Greyscale Erosion



Greyscale Erosion

- The function applies the rule to the input pixel's neighborhood and uses the *minimum* value of all the pixels in the neighborhood as the value of the corresponding pixel in the output image.
- Grayscale erosion results in a decreased overall brightness, expanded dark regions, and loss of small bright details.

Grayscale Erosion Example

grayscale image

structuring element

0	1	0
1	1	1
0	1	0

0	0	1	0	0
0	1	1	1	0
1	1	1	1	1
0	1	1	1	0
0	0	1	0	0

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

eroded image

Exercise 4

1. Read in the rice image:

```
I = imread('rice.png');
```

2. Create a disk structuring element with radius 2:

```
se = strel('disk',2);
```

3. Erode the grayscale image using the structuring element:

```
I_e = imerode(I,se);
```

4. Display the eroded image:

```
imshow(I_e)
```

5. repeat steps 2, 3 and 4 using:
   - a disk structuring element with radius 4
   - a disk structuring element with radius 8
   - a disk structuring element with radius 16

Operations Based on Dilation and Erosion

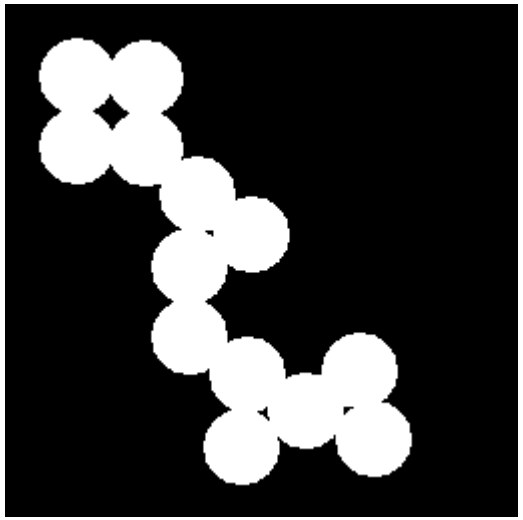
- Dilation and erosion are often used in combination to implement image processing operations:
  - Opening
  - Closing
  - Skeletonization
  - Top-Hat
  - Bottom-Hat

Opening

- The opening operation erodes an image and then dilates the eroded image, using the same structuring element for both operations.
- Morphological opening is useful for removing small objects from an image while preserving the shape and size of larger objects in the image.

Binary Opening Example

binary image



noisy image

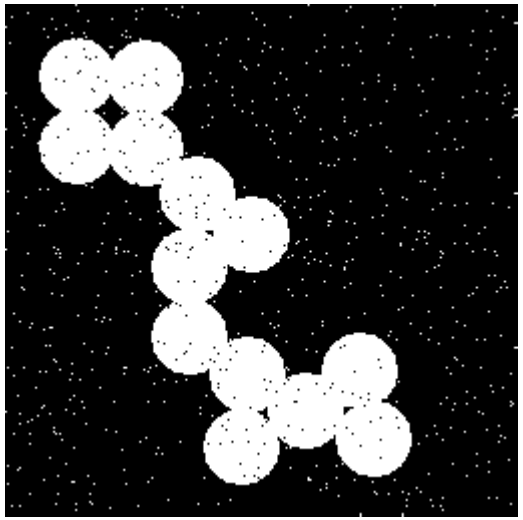
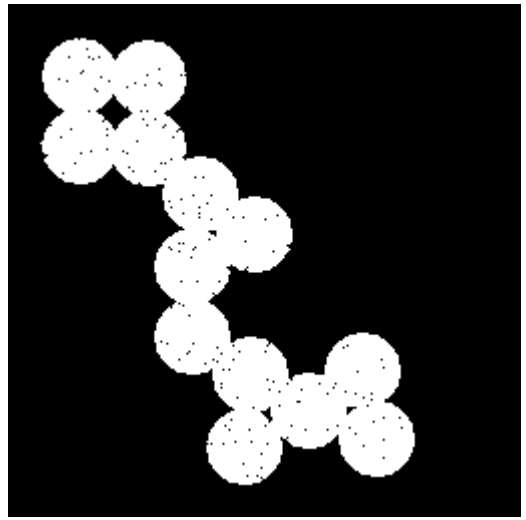
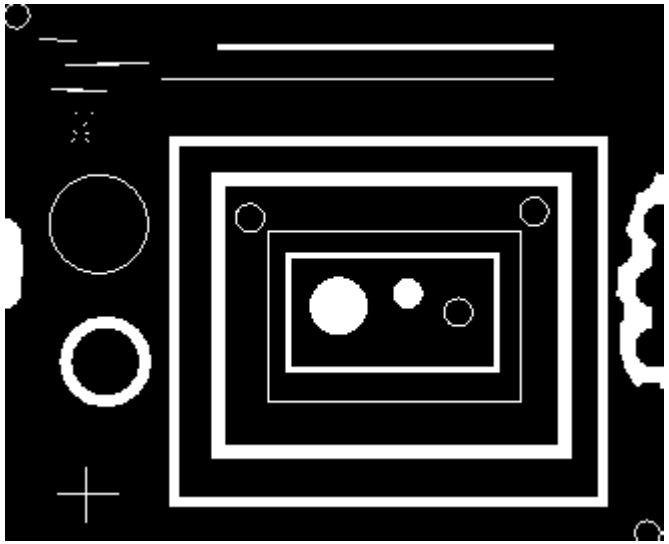


image after opening



Exercise 5

1. Use the `imopen` function to remove the thin lines from the image `'test_pattern.png'` shown below.



Grayscale Opening Example

grayscale image

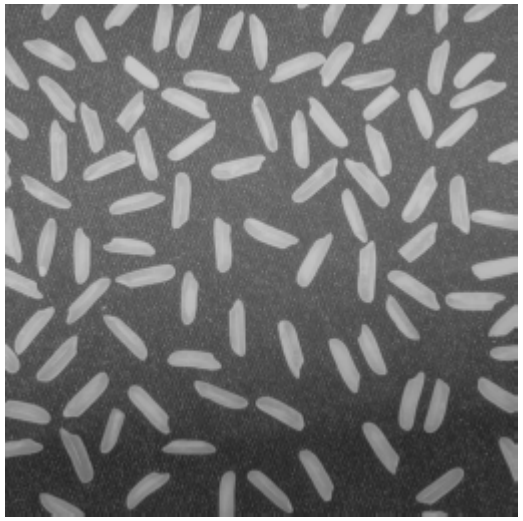
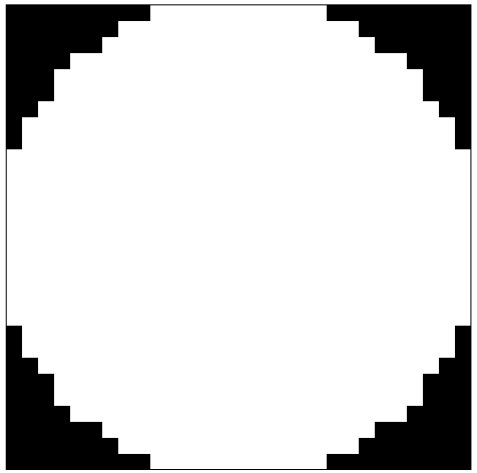


image after opening

structuring element (29 x 29)



Exercise 6

1. Read in the rice image:

```
I = imread('rice.png');
```

2. Create a disk structuring element with radius 2:

```
se = strel('disk',2);
```

3. Perform grayscale opening on the image using the structuring element:

```
I_o = imopen(I,se);
```

4. Display the eroded image:

```
imshow(I_o)
```

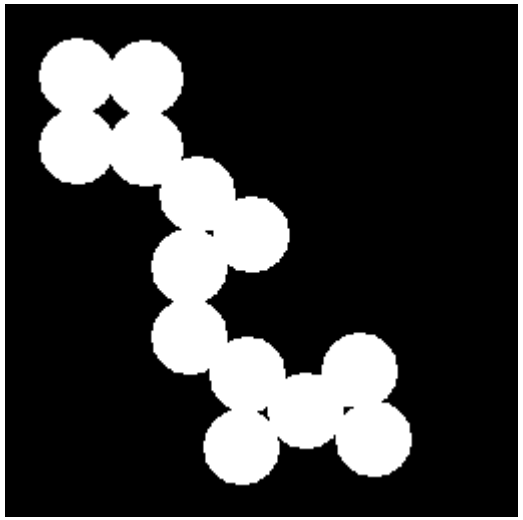
5. repeat steps 2, 3 and 4 using:
   - a disk structuring element with radius 4
   - a disk structuring element with radius 8
   - a disk structuring element with radius 16

Closing

- The closing operation dilates an image and then erodes the dilated image, using the same structuring element for both operations.
- Morphological closing is useful for filling small holes from an image while preserving the shape and size of the objects in the image.

Binary Closing Example

binary image



noisy image

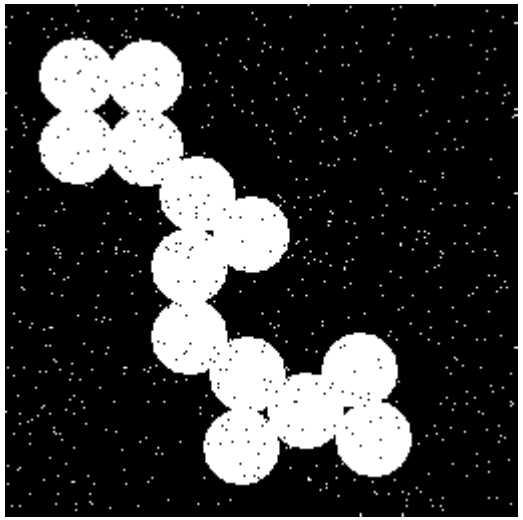
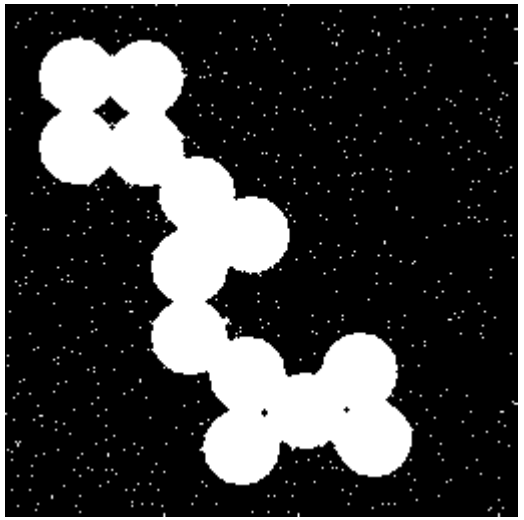


image after closing

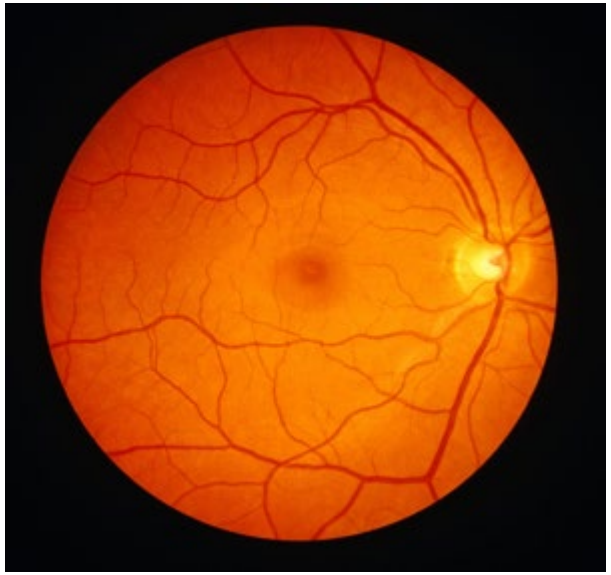


Exercise 7

1. Use the `imclose` function to remove the gap between the two lines shown in the `'line_gap.png'` image shown below.

Grayscale Closing Example

colour image



grayscale image

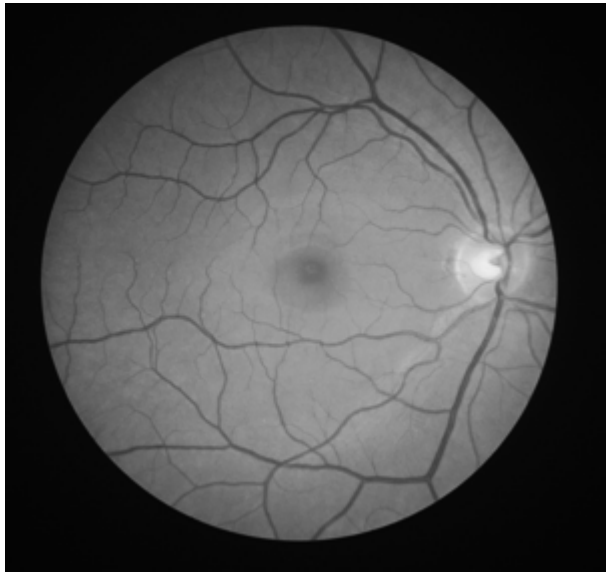
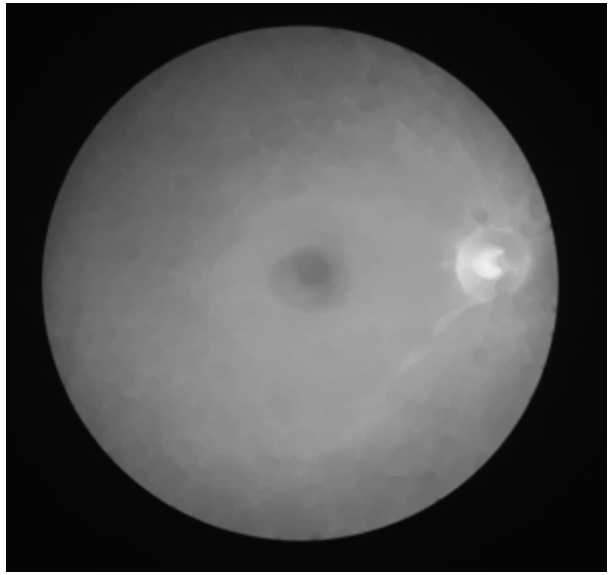
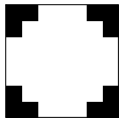


image after closing



structuring element (7 x 7)



Exercise 8

1. Read in the retina image:

```
RGB = imread('retina.jpg');
```

2. Convert it to a grayscale image:

```
I = rgb2gray(RGB);
```

3. Create a disk structuring element with radius 1:

```
se = strel('disk',1);
```

4. Close the grayscale image using the structuring element:

```
I_c = imclose(I,se);
```

5. Display the dilated image:

```
imshow(I_c)
```

6. repeat steps 3, 4 and 5 using:
   - a disk structuring element with radius 2
   - a disk structuring element with radius 4
   - a disk structuring element with radius 8

Top-Hat

- The top-hat transform opens an image, then subtracts the opened image from the original image.
- The top-hat transform can be used to enhance contrast in a grayscale image with nonuniform illumination.
- The transform can also isolate small bright objects in an image.

Grayscale Top-Hat Example

grayscale image

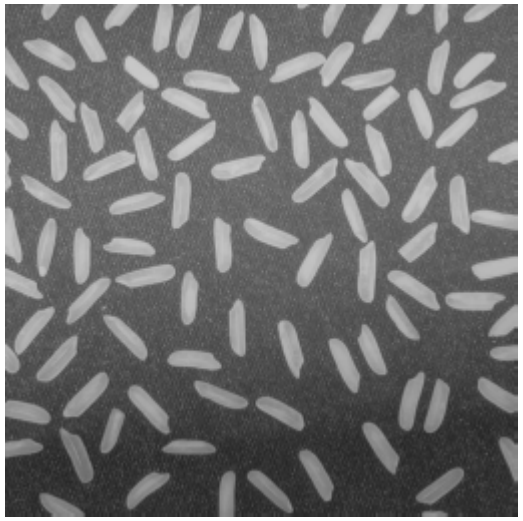
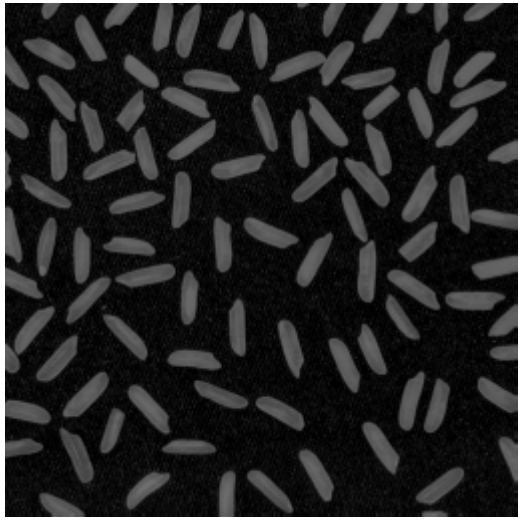
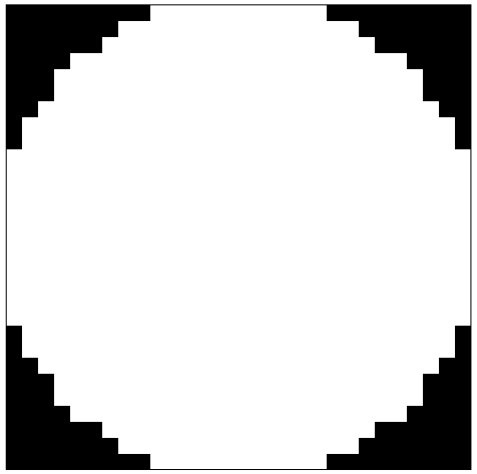


image after top-hat



structuring element (29 x 29)



Exercise 9

1. Read in the rice image:

```
I = imread('rice.png');
```

2. Create a disk structuring element with radius 2:

```
se = strel('disk',2);
```

3. Perform the grayscale top-hat operation on the image using the structuring element:

```
I_th = imtophat(I,se);
```

4. Display the eroded image:

```
imshow(I_th)
```

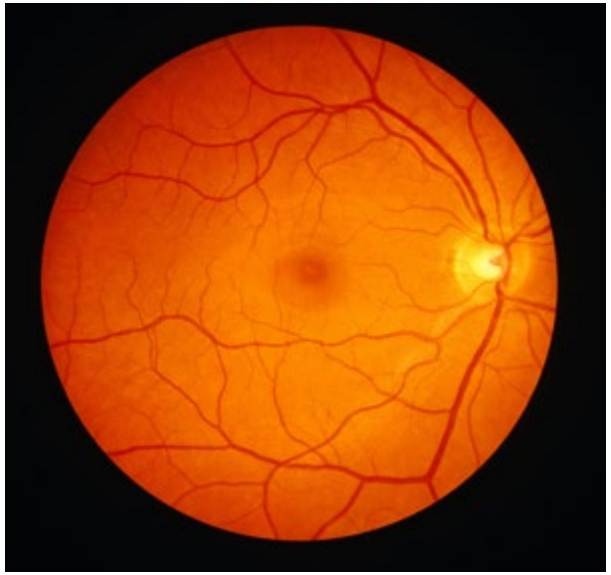
5. repeat steps 2, 3 and 4 using:
   - a disk structuring element with radius 4
   - a disk structuring element with radius 8
   - a disk structuring element with radius 16

Bottom-Hat

- The bottom-hat transform closes an image, then subtracts the original image form the closed image.
- The bottom-hat transform can be used to find intensity troughs in a grayscale image.

Grayscale Bottom-Hat Example

colour image



grayscale image

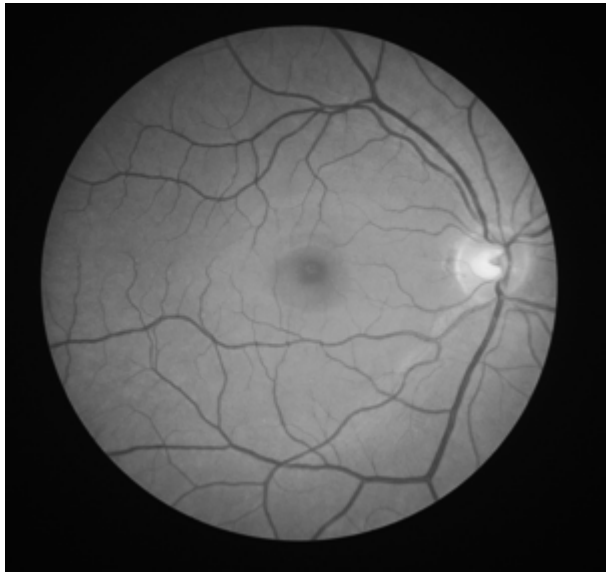
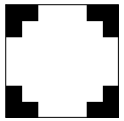


image after bottom-hat

structuring element (7 x 7)



Exercise 10

1. Read in the retina image:

```
RGB = imread('retina.jpg');
```

2. Convert it to a grayscale image:

```
I = rgb2gray(RGB);
```

3. Create a disk structuring element with radius 1:

```
se = strel('disk',1);
```

4. Perform the grayscale bottom-hat operation on the image using the structuring element:

```
I_bh = imbothat(I,se);
```

5. Display the dilated image:

```
colormap(gray(256))
imagesc(I_bh)
```

6. repeat steps 3, 4 and 5 using:
   - a disk structuring element with radius 2
   - a disk structuring element with radius 4
   - a disk structuring element with radius 8

Skeletonization

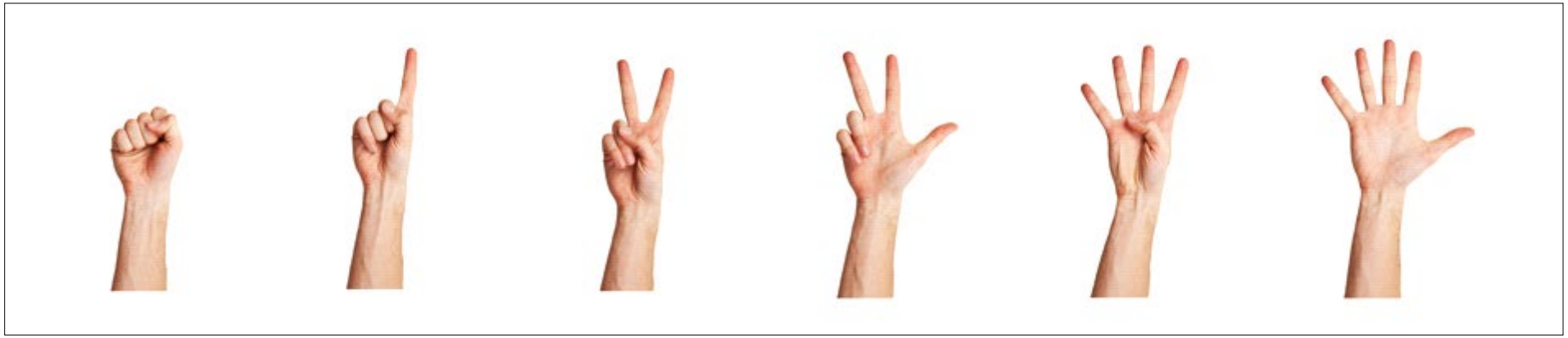
- The process of skeletonization erodes all objects to centerlines without changing the essential structure of the objects, such as the existence of holes and branches.
- Skeletonization transforms objects in an image into a set of lines that run roughly down the center of each object.
- The size of the input object is maintained and the endpoints of the skeleton extend all the way to the edges of the input object.

Skeletonization

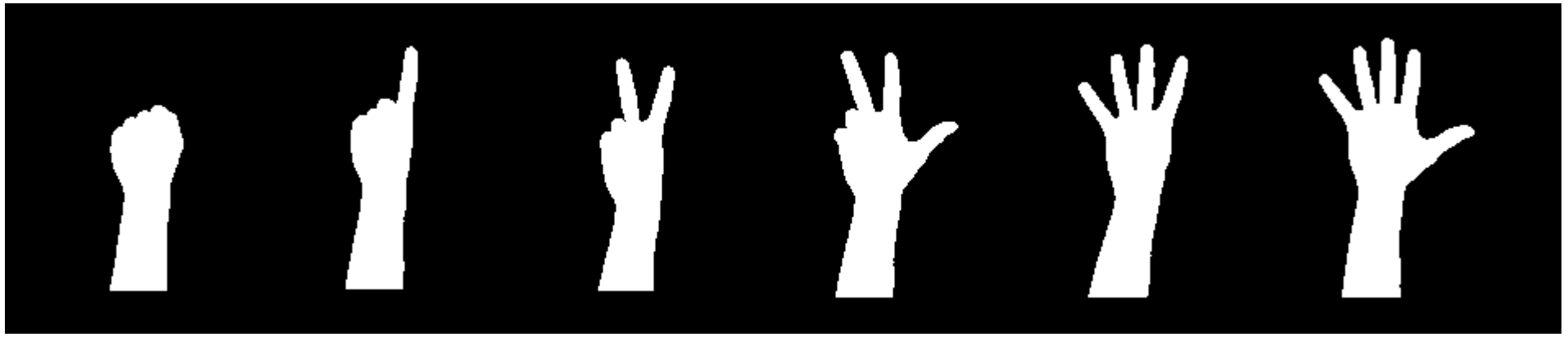
- Skeletonization provides a compact yet effective representation of 2-D and 3-D objects.
- It is useful in many low- and high-level image-related tasks including object representation, retrieval, manipulation, matching, registration, tracking, recognition, and compression.
- It also facilitates efficient characterization of topology, geometry, scale, and other related local properties of an object.

Skeletonization Example

colour image



binary image



Skeletonization Example

after skeletonization