National University of Science and Technology

School of Electrical Engineering and Computer Science

Department of Software Engineering

EE433: Digital Image Processing

Class: BESE-5

Lab 9: Morphological Image Processing - 1

Date: 14th November, 2017

Time: 2pm-5pm

Instructor: Dr. Muhammad Moazam Fraz

Lab Engineer: Ms Iram Tariq Bhatti

Course Learning Outcomes (CLOs)			
Upon completion of the course, students should demonstrate the ability to:		**PLO Mapping**	**BT Level***
CLO 1	Understanding the fundamentals and basic concepts of image processing	PLO 1	C2
CLO 2	Analyze images using mathematical transformations and operations	PLO 2	C4
CLO 3	Develop solutions by using modern tools to solve practical problems.	PLO 5	C5
	* BT= Bloom's Taxonomy, C=Cognitive domain, P=Psychomotor domain, A= Affective domain		

Table of Contents

Learning Outcome

CLO 3: Develop solutions by using modern tools (Matlab) to solve practical problems.

Goal

The goal of this lab is to learn how to implement the basic binary morphological operations.

Objectives

- Learn how to dilate an image using the imdilate function.
- Learn how to erode an image using the imerode function.
- Learn how to open an image using the imopen function.
- Learn how to close an image using the imclose function.
- Explore the hit-or-miss transformation using the bwhitmiss function.
- Learn how to perform boundary extraction.
- Explore the bwperim function.
- Learn how to fill object holes using the imfill function.

Tools/Software Requirement

MATLAB 201x / Python 2 or 3 with associated libraries.

Morphological Operations: An Overview

Mathematical morphology is a tool for extracting image components useful in the representation and description of region shape, such as boundaries, skeletons and convex hulls. The language of mathematical morphology is set theory, and as such it can apply directly to binary (two-level) images: a point is either in the set (a pixel is set, or put to foreground) or it isn't (a pixel is reset, or put to background), and the usual set operators (intersection, union, inclusion, complement) can be applied to them.

Basic operations in mathematical morphology operate on two sets: the first one is the *image*, and the second one is the *structuring element* (sometimes also called the *kernel*, although this terminology is generally reserved for convolutions). The structuring element used in practice is generally much smaller than the image, often a 3x3 matrix. It is possible, however, to make a generalization to grey level images. Erosion and dilation are two basic operators in mathematical morphology

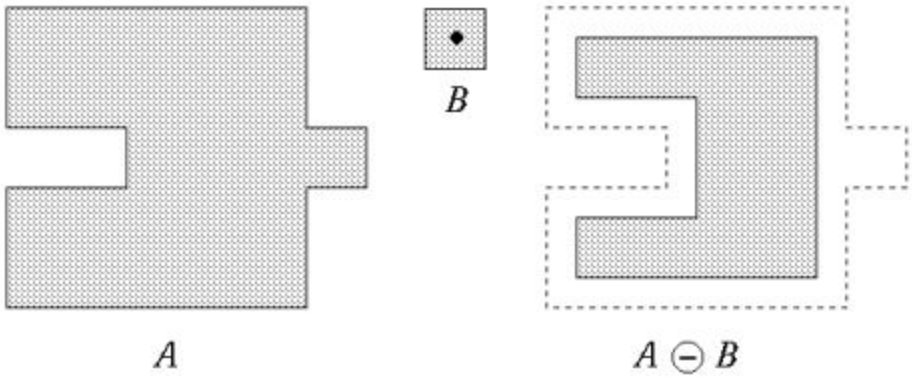
Erosion:

The basic effect of erosion operator on a binary image is to erode away the boundaries of foreground pixels (usually the white pixels). Thus areas of foreground pixels shrink in size, and "holes" within those areas become larger. Let foreground pixels be represented by logical 1's, and background pixels by logical 0's. As a practical example, we take a 3x3 matrix of logical 1's, with the middle point chosen as the origin of the set is used as the structuring element *B*.

To compute the erosion of a binary input image by this structuring element, we consider each of the foreground pixels in the input image in turn. For each input pixel we superimpose the structuring element on top of the input image so that the origin of the structuring element coincides with the input pixel coordinates.

1. If the input pixel is set to foreground and all its 8 neighbors are also set to foreground, then the pixel remains set to foreground.
2. If the input pixel is set to foreground, but at least one of its 8 neighbors is not, the pixel is set to background.
3. Input pixels set to background remain such.

With the structuring element chosen as above, the effect of this operation is to remove any foreground pixel that is not completely surrounded by other foreground pixels, assuming 8-connectedness. We can also see that this operation can be performed on binary images simply by applying a logical AND function.

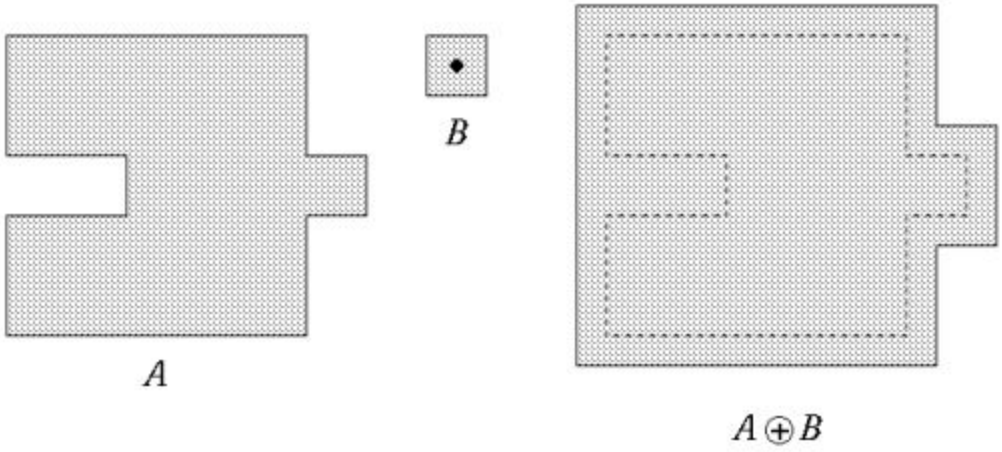


Dilation

The dilation is the other of the two basic operators in mathematical morphology. The basic effect of dilation on binary images is to enlarge the areas of foreground pixels (*i.e.* white pixels) at their borders. The areas of foreground pixels thus grow in size, while the background "holes" within them shrink.

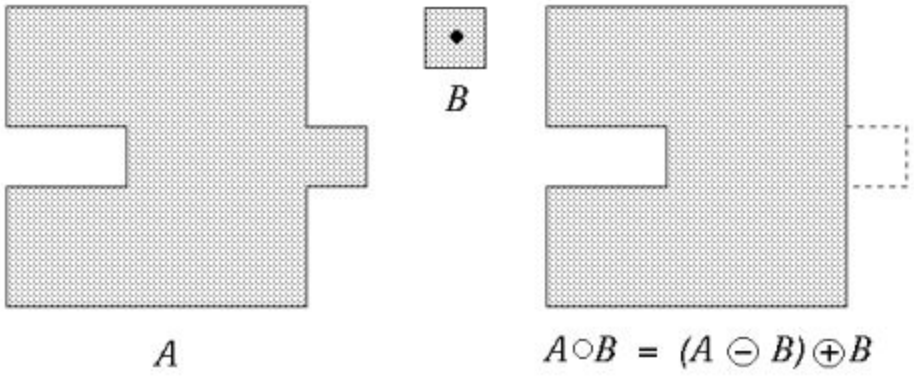
Again let us take a 3x3 matrix for the structuring element, with the center pixel used as the origin of the set. *B*, then the dilation can be performed using the logical OR function:

1. If the pixel is set to foreground, it remains such.
2. If the pixel is set to background, but at least one of its eight neighbor's is set to foreground, the pixel is converted to foreground.
3. If the pixel is set to background and none of its eight neighbor's is set to foreground, the pixel remains set to background.

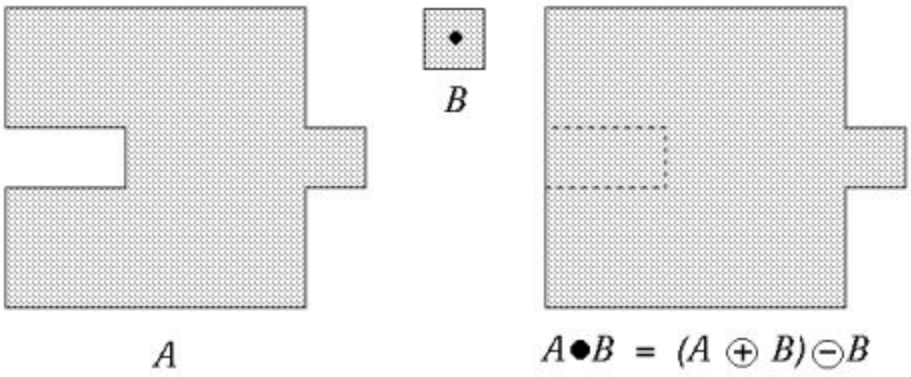


Opening and Closing

The **opening** is a composite operator, constructed from the two basic operators described above. Opening of set *A* by set *B* is achieved by first the eroding set *A* by *B*, then dilating the resulting set by *B*.



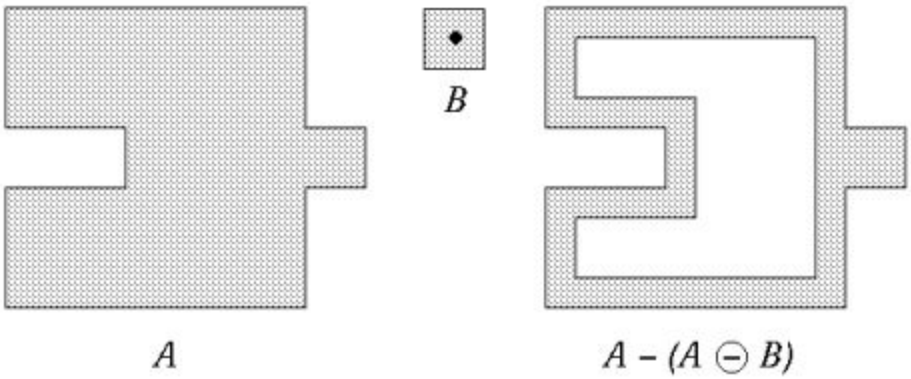
The **closing**, like opening, is also a composite operator. The closing of set *A* by set *B* is achieved by first dilating of set *A* by *B*, then eroding the resulting set by *B*.



On the left side there is the original image, on the right side is the closed image.

Boundary Extraction

The boundary of set *A* can be found by first eroding *A* by *B*, then taking the set difference between the original *A* and the eroded *A*.



Here is an example of binary image outlining:

Generalization to Grayscale Images

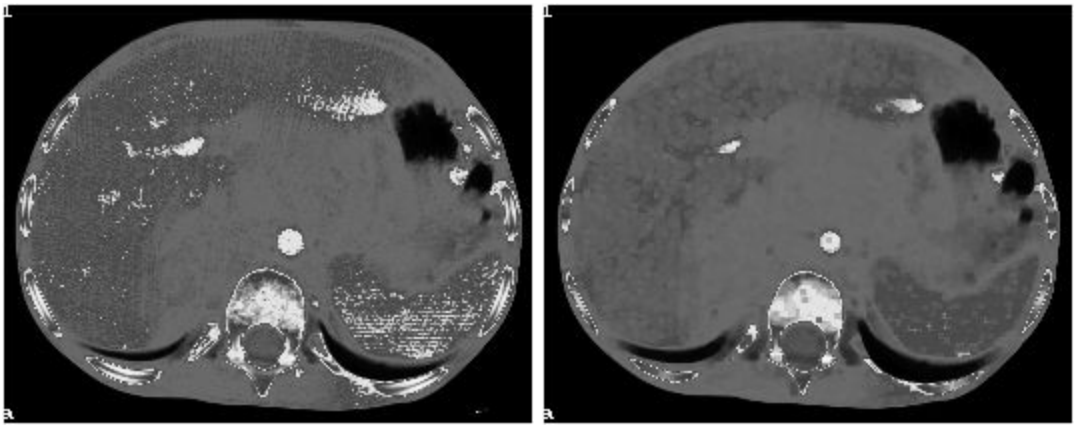
The ideas above can be extended to grayscale images as well. In grayscale images, each pixel can have the value in a certain range, *i.e.* 0 to 255, with 0 representing black and 255 representing white.

Grayscale Erosion and Dilation

Grayscale **erosion** is analogous to its binary counterpart. Grayscale erosion darkens small bright areas, and very small bright areas like noise spikes or small spurs might be totally removed.

Working with a 3x3 structuring element as above, grayscale erosion is implemented as follows:

- the numerical values of the point and its eight neighbors are evaluated
- minimal value of these 9 values is found
- the new value of the point is set to the minimal value

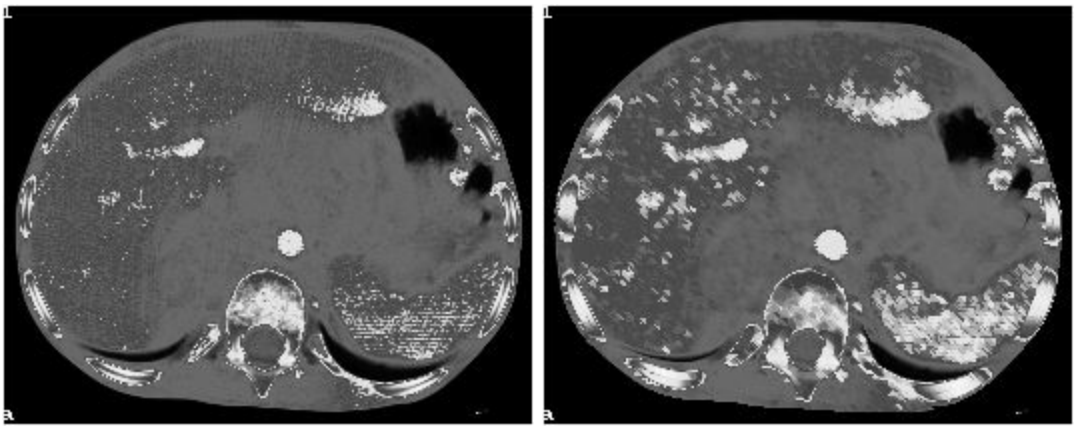


On the left side there is the original image, on the right side is the eroded image.

Grayscale **dilation** is analogous to its binary counterpart. Grayscale dilation brightens small dark areas, and very small dark "holes" might be totally removed.

Working with a 3x3 structuring element as above, grayscale dilation is implemented as follows:

- the numerical values of the point and its eight neighbors are evaluated
- maximal value of these 9 values is found
- the new value of the point is set to the maximal value

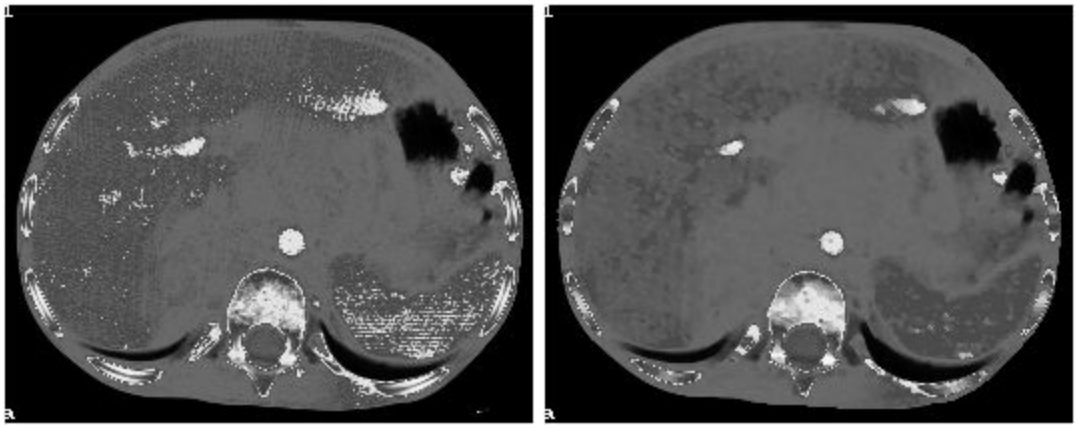


On the left side there is the original image, on the right side is the dilated image.

Grayscale Opening and Closing

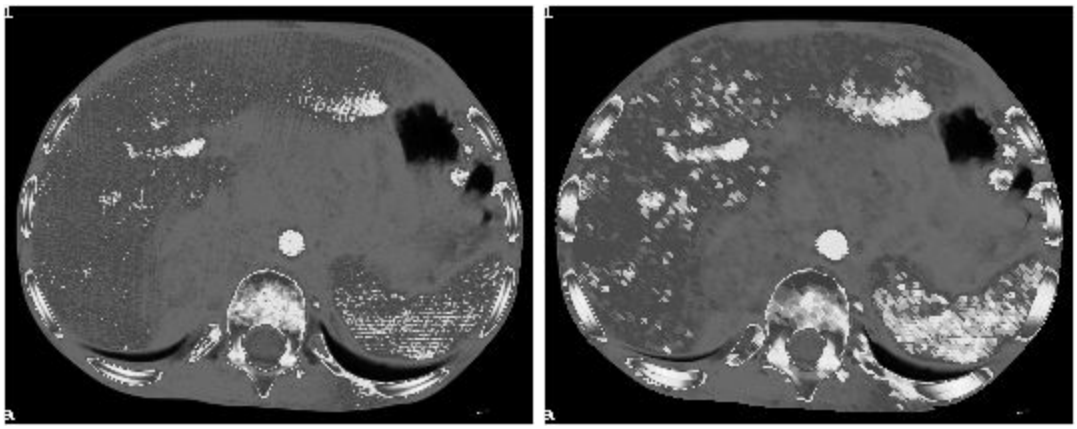
Grayscale opening is, like its binary counterpart, achieved by first eroding the image with the structuring element, and then dilating the resulting image by the structuring element. The

process darkens small bright areas, and may entirely remove very small bright spots like noise spikes.



On the left side there is the original image, on the right side is the opened image.

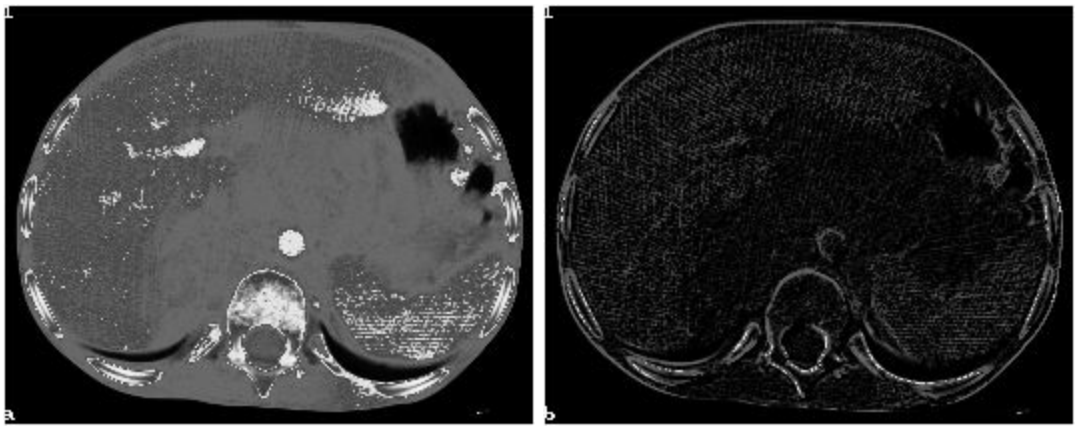
Grayscale closing is, like its binary counterpart, achieved by first dilating the image with the structuring element, and then eroding the resulting image by the structuring element. The process brighteness small dark areas, and may entirely remove very small dark "holes".



On the left side there is the original image, on the right side is the closed image.

Top-Hat Transformations

The **Top-Hat Transform** or peak detector is another composite operation: the image opened by the structuring element is subtracted from the original image. The brightest spots on the original image are highlighted using this transformation.

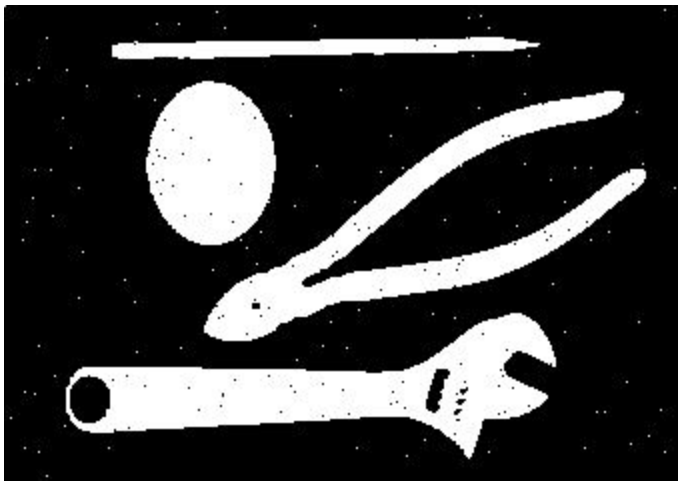


On the left side there is the original image, on the right side is the image after top-hat transform.

Exercises

Problem 1: Binary Morphology

Consider the following noisy binary image. To obtain it, you should right-click on it and save the file to your local directory.



Task 1: Apply erosion, dilation, opening, and closing by a 3x3 "cross" structuring element. Are any of the results satisfying? Let B denote this S.E. Notice how the shape and size of B is revealed in both the dilated and eroded images. The structuring element can be obtained with the command B = strel('diamond',1).

Hint: you may use the functions imerode(), imdilate(), imopen(), and imclose() to accomplish this.

Task 2: A better result can be obtained by applying alternating sequential filters. Apply the open-close and close-open filters by B to the noisy image. What do you observe?

Task 3: The result is now quite good, but still not completely satisfying (notice the degradation introduced to the wrench). This kind of degradation is typical of "structural" morphological filters; it can be avoided by using reconstructive morphological filters. A simple solution employs the "close-holes" operation (which is a reconstructive filter). Let CH denote the image obtained with application of the close-holes operator. If we subtract the original image from this image, and clean up the noise in the result with a simple opening by B, we get an image H with all the tools holes. Now, going back to the CH image, it is quite noisy, but it only has "positive" noise (since all negative noise was eliminated by the close-holes filter). This noise can be removed by a simple opening by B. Now all that is left to do is to impose ("punch")

the holes by subtracting the saved H image from this. Display all the images and comment on the result.

Hint: You can implement the close-holes operator by the process or equivalently on page 660 of Gonzalez; however you may also take a shortcut and use the command imfill(I,'holes').

Task 4: Yet another way to solve this problem is using reconstruction directly. Note that closing removes the negative noise, while opening removes the positive noise. Therefore, it seems reasonable that using the opening as the marker and the closing as the mask of a reconstruction should do the job. Verify if this is indeed the case, assuming 4-connectivity for the reconstruction. This solution is simpler, but is it identical to the solution in the previous item?

Hint: Reconstruction is implemented by the command imreconstruct().

Deliverables

Perform above mentioned practice tasks in Matlab or Python.

Just make a new folder named as DIP-Lab9 in existing private repository and then add the teacher as a collaborator. All the code must be in runnable format in order to get the credit.

1. A file with commented source code representing the work accomplished for this lab.
2. All files should contain author in the comments at the top of the file.