CS5680/CS6680 – Fall Semester 2018
Assignment 5 – Morphological Operations
Due: 11:59 p.m., Tuesday, October 30, 2018
**Total Points: 45 points**

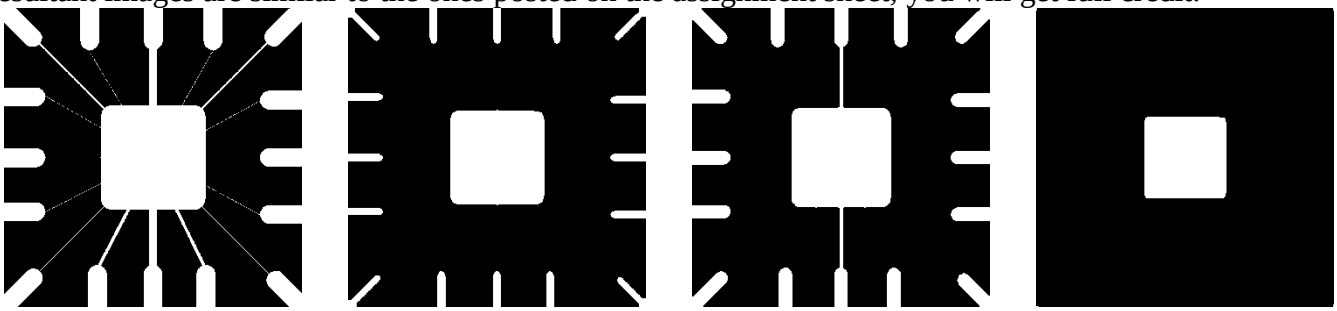
**General Assignment Instructions:** The same as the previous assignments.

**Problem I: Problem Solving Using Morphological Operations [Total: 20 points]**
**Note: You can call Matlab built-in morphological operations to solve all sub-problems of Problem I.**

**1. [2 points]**
Load in *City.jpg* image. Apply the morphological gradient operation $g = (f \oplus b) - (f \ominus b)$ where $f$ is the original image and $b$ is the 3×3 square structuring element. Display the resultant image in figure 1.

**2. [3 points]**
Load in *SmallSquares.tif* image. Apply the appropriate morphological operation(s) on the original image to locate foreground pixels that have east and north neighbors and that have no north-west, west, southwest, south, or south-east neighbors. Display the final resultant image in figure 2. Use the Matlab console to show the number of foreground pixels that satisfy the above conditions.

**3. [5 points]**
Load in *Wirebond.tif* image. Apply appropriate morphological operation(s) on the **original image** to obtain the following three desired images (i.e., (b), (c), and (d)), respectively. Display the three resultant images in figure 3 with appropriate titles. Here, I list the original image for your reference so you can visually compare the differences between the original image and the expected resultant images. As long as your resultant images are similar to the ones posted on the assignment sheet, you will get full credit.



(a) Original image (b) Desired image 1 (c) Desired image 2 (d) Desired image 3

**4. [5 points]**
Load in *Shapes.tif* image. Apply appropriate morphological operation(s) on the **original image** to obtain the following three desired images (i.e., (b), (c), and (d)), respectively. Display the three resultant images in figure 4 with appropriate titles. Here, I list the original image for your reference so you can visually compare the differences between the original image and the expected resultant images. As long as your resultant images are similar to the ones posted on the assignment sheet, you will get full credit.

(a) Original image (b) Desired image 1 (c) Desired image 2 (d) Desired image 3

**5. [5 points]**
Load in *Dowels.tif* image. Apply an open-close operation on *Dowels.tif* using a disk structuring element of radius 5 (i.e., an open operation followed by a close operation is applied to the original image using the same structuring element). Apply a close-open operation on *Dowel.tif* using a disk structuring element of radius 5 (i.e., a close operation followed by an open operation is applied to the original image using the same structuring element). Display two resultant images in figure 5 side-by-side.

Apply a series of open-close operations on *Dowel.tif* using a series of structuring elements of increasing size (i.e., a disk structuring element of radius of 2, 3, 4, and 5). Similarly, apply a series of close-open operations on *Dowel.tif* using a series of structuring elements of increasing size (i.e., a disk structuring element of radius of 2, 3, 4, and 5). Display two resultant images in figure 6 side-by-side. You have to use "loop" to include the series of open-close operations or close-open operations. In other words, each loop will perform one open-close or close-open operation with a disk structuring element of a certain radius.

**Problem II: Applications of Morphological Operations [Total: 25 points]**

A preprocessing step in an application of microscopy is to isolate individual round particles from similar particles that overlap in a group of two or more particles (see image "**Ball.tif**"). Assuming that **all particles are of the same size**, use the "**extraction of connected components**" morphological operation (pp. 536–539 of the second edition of the Digital Image Processing textbook or pp. 645–647 of the third edition of the Digital Image Processing textbook) to solve the following sub-problems. Here is the basic idea of the "extraction of connected components" algorithm:
Let $Y$ be a connected component contained in an image $A$ and assume that a point $p$ of $Y$ is known. Then the following iterative expression yields all the elements of $Y$.

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1, 2, 3, .....$$

where $X_0 = p$ and $B$ is a suitable structuring element.
The algorithm terminates at iteration step $k$ if $X_k = X_{k-1}$. The algorithm converges to $Y = X_k$.

**1. [8 points]**
Using the idea of "extraction of connected components" mentioned at the beginning of the problem, implement a **FindComponentLabels** function to label connected objects (i.e., connected components with the value of 1's) in a binary image. The prototype of this function should be:

**function [labelIm, num] = FindComponentLabels(im, se)**

where im is the original binary image, se is the structuring element, labelIm is the labeled image which contains the labeled connected objects **(Please sequentially label the connected objects with 1 being the starting label)**, and num is the number of connected objects (i.e., **the largest label**) found in the binary image. It should be noted that different connected objects have different labels and the labels for all the elements in a connected object are the same. In addition, the smallest label for a connected object is 1 and the largest label for a connected object is num.

Call this function to label the connected particles and return the total number of connected particles in a given image **Ball.tif.** Please use the Matlab console to display the total number of connected particles in **Ball.tif** found by **FindComponentLabels**. Display these connected objects in figure 7 using appropriate visible gray-level intensities or color intensities.

**2. [2 points]**
Call an appropriate Matlab built-in function, which accomplishes the same functionality as the **FindComponentLabels** function, to label connected particles and return the total number of connected

particles. Use the Matlab console to display the number of connected particles in **Ball.tif** found by the Matlab built-in function. Display these connected particles in figure 8 using appropriate visible gray-level intensities or color intensities. Note: *Depending on the implementation, FindComponentLabels function and Matlab built-in function may label the same connected particles using different labels. However, the total number of connected particles should be the same.*

**3. [5 points]**
Produce an image **A** containing only **connected particles** not residing on the border of the image. Display the original image and the image **A** side-by-side in figure 9 with appropriate titles. Use the Matlab console to show the number of connected particles not residing on the border. You have to write your own solution to this problem without calling Matlab built-in function.

**4. [2 points]**
Call an appropriate Matlab built-in function to solve the previous problem (i.e., Problem II.3). Use the Matlab console to show the number of connected particles not residing on the border. Display the image A and the result produced by the Matlab built-in function side-by-side in figure 10 with appropriate titles.

**5. [8 points]**
Produce an image **B** containing only visually **non-overlapping connected particles** not residing on the border of the image. Display the original image and the image **B** side-by-side in figure 11 with appropriate titles. Use the Matlab console to show the number of the non-overlapping connected particles that do not reside on the border. Hint: Since all particles are of the same size, you may need to design a strategy to estimate the size of the individual particle to solve the problem. **Please DO NOT hard code any values (e.g., 10, 20, etc.) to solve the problem.**