LAB # 02: Connected Component Analysis

Lab Objective:

The objective of this lab is to introduce the student with some transformation especially with respect to image processing and perform connected component labelling in images and to get an understanding of intensity level resolution.

Lab Description:

Intensity level resolution defines the resolution at bit level i.e. how many bits are used to represent a pixel value. The more bits we have per pixel, the more levels there are. For example, a grayscale image has 256 different levels because for each pixel value we use 8 bits to store the value. If the bits per pixel are decreased to 4, then the maximum levels that we can have is 16. Similarly, if only 1 bit is used for it then we can have only two levels (or a binary image).

To change the bit level resolution, different levels (or intensities) can be grouped together to form new levels. For example, for a 3 bits/pixel image:

0	1	2	3	4	5	6	7

0	1	2	3

0	1

Connected Component Analysis

Connected Component Analysis or Labelling enables us to detect different objects from a binary image. Once different objects have been detected, we can perform a number of operations on them: from counting the number of total objects to counting the number of objects that are similar, from finding out the biggest object of the bunch to finding out the smallest and from finding out the closest pair of objects to finding out the farthest etc.

Connected Component labelling procedure is as follows:

- Process the image from left to right, top to bottom:

  If the next pixel to process is 1

i.) If only one from top or left is 1, copy its label.

ii.) If both top and left are one and have the same label, copy it.

iii.) If top and left they have different labels

− Copy the smaller label

− Update the equivalence table.

iv.) Otherwise, assign a new label.

- Re-label with the smallest of equivalent labels

Video explanation: https://www.youtube.com/watch?v=ticZclUYy88

Some Useful Commands:

i) arr = np.array([[1, 2, 3] , [6, 5, 4]])

arr + 2 will add 2 in each element of arr

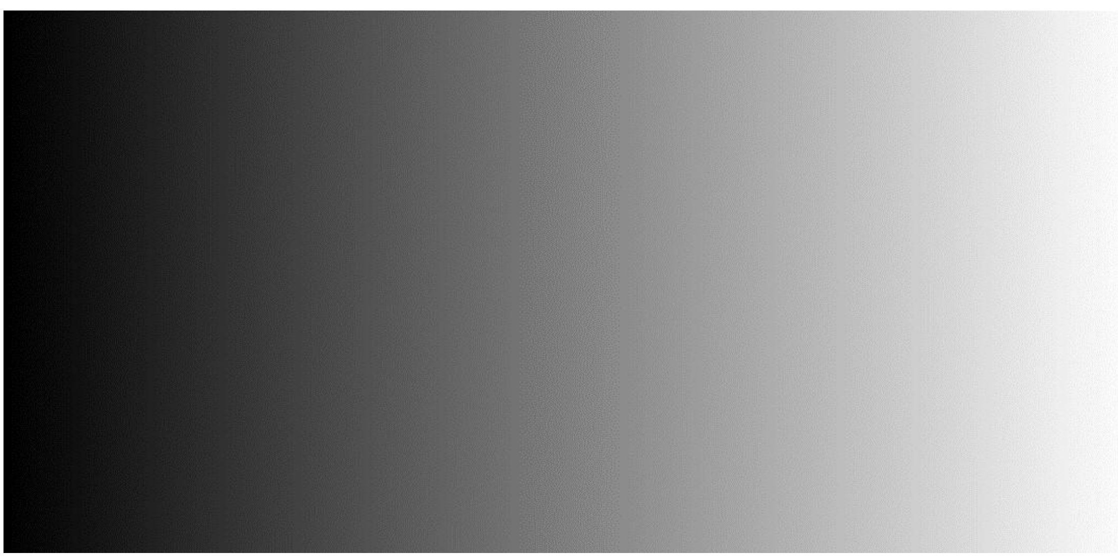
ii) arr = np.array([[1, 2, 3] , [6, 5, 4]])

arr == 2 will return following bolean array

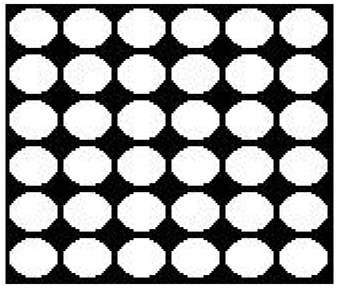
array([[False, True, False] , [False, False, False]])

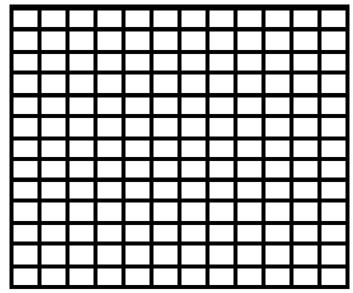
<u>Lab Tasks:</u>

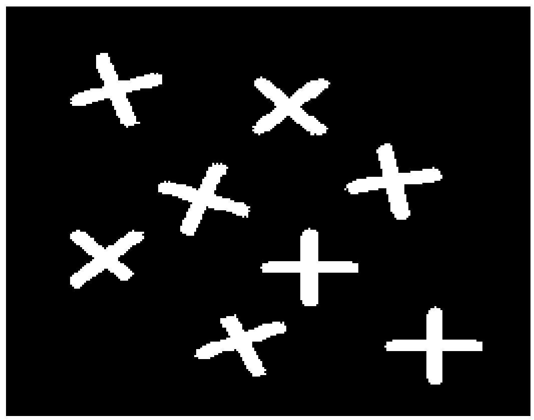
1. Read a grayscale image (given below) and convert the image to 16 levels, then to 4 levels and finally to 1. Display all four images.



2. For the images given below (also available with the lab handout), apply the connected component labelling using 4 connectivity and count the total number of objects in the list. (HINT: In the image given here, the background (black portion) has a numeric value of 1 while the white objects have a numeric value of 255.)

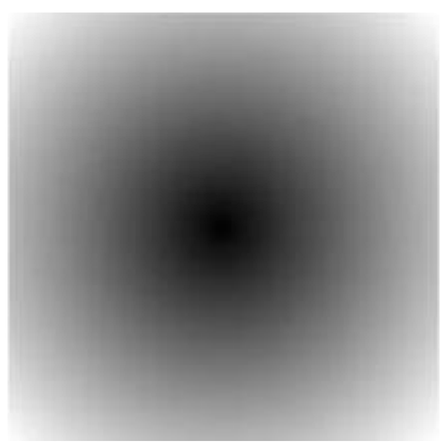






<u>Home Task:</u>

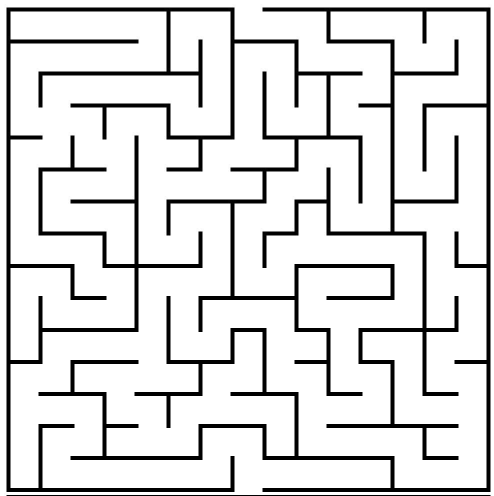
A **Distance map** can be created by measuring the Euclidian distance of every pixel (position in x and y) from the center and then assigning that value to the pixel for which the Euclidian distance has been calculated. A distance map is shown below:



The following are your home tasks:

1. Create a Euclidean distance map for a 501x501 empty image. Convert this image to 16 levels, then to 4 levels and finally to 1.

2. Perform connected component labeling on the masked image with 4-connectivity. Only color the path from start to finish. You have find the correct path (from start to finish) and only highlight that path.



Maze