**Assignment-1**

**1. Resampling**

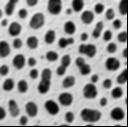
1. Taking image, resize factors(fx,fy) and interpolation method as inputs through command line arguments.

2. Depending upon the resize factors (fx,fy) we will resize the input image.

3. For an input image we will calculate number of rows and columns by “image.shape “ .Then by multiplying with resize factors we will obtain the number of rows and columns for an output image, with this dimensions we will create an empty grid and assign values according to interpolation method.

4. In nearest neighbor interpolation, for a particular pixel grid we will compare with the input image and find the nearest pixel grid in the input image and assign that value. We will apply this for all the values and obtain the output resized image

5. In bilinear interpolation, we will calculate the weighted average of input neighbor pixels for a given output image pixel. First we will apply linear interpolation for all neighboring pixels separately with possible pairs other than diagnols and then compute the weighted average and will assign that value to the output image.



resized image for (fx,fy) = (0.5,0.5)

**2a. Region counting:**

1. For the given input image calculate frequency at each level.

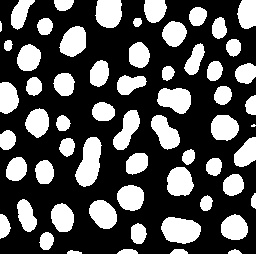
2. Plot these values along with frequencies which results in the histogram.

3. Assuming that the distribution has two peaks, first assigned a random value and then calculated the two means of the distribution.

4. The average of the two mean values will give the threshold.

5. Assuming foreground takes value 0 and background takes a value 255, I assigned all pixels having value less than threshold as 255 and the rest as 0. This gives a binary image of the input image.

**if** (image[i, j]) < a:  
 out\_img[i, j] = 255  
**else**:  
 out\_img[i, j] = 0



**2b. Blob coloring**

1. This takes the binary image generated from the previous step as input.

2. I have initialized an array R for region coloring and then filled values into it based on the following conditions, where I have considered and compared a particular pixel with its left, topleft,top,bottom and bottom left pixels

some of the conditions are as follows

**if** Input[i, j] == 255 **and** Input[i, j - 1] == 0 **and** Input[i - 1, j - 1] == 0 **and** Input[i - 1, j] == 0 **and** Input[  
 i - 1, j + 1] == 0:  
 R[i, j] = total  
 total = total + 1  
**if** Input[i, j] == 255 **and** Input[i, j - 1] == 0 **and** Input[i - 1, j - 1] == 0 **and** Input[i - 1, j] == 0 **and** Input[  
 i - 1, j + 1] == 255:  
 R[i, j] = R[i - 1, j + 1]  
**if** Input[i, j] == 255 **and** Input[i, j - 1] == 0 **and** Input[i - 1, j - 1] == 0 **and** Input[i - 1, j] == 255 **and** Input[  
 i - 1, j + 1] == 0:  
 R[i, j] = R[i - 1, j]  
**if** Input[i, j] == 255 **and** Input[i, j - 1] == 0 **and** Input[i - 1, j - 1] == 255 **and** Input[i - 1, j] == 0 **and** Input[  
 i - 1, j + 1] == 0:  
 R[i, j] = R[i - 1, j - 1]

3. By applying all the conditions I am assigning values to R and at the same time keeping a track of number of regions with a counter with the loop.

4. This returns the list of regions.

**2c. cell counting**

1. From the regions obtained from blob coloring, I have calculated the centroids and area for each blob

2. I have displayed those values with \* symbol which represents the centroid with cv2.putText() function

3. This gives us all the regions centroids and areas and also an output image with symbols displayed on it

