**Morphological Transformations on Images with OpenCV:**

What are Morphological Transformations?

Morphological Transformations are the set of operations based on the morphology i.e., shape of image.

* Generally performed on binary images.
* Structuring element: It is defined as a small shape( or matrix )which is used to traverse the whole image while comparing every pixel with its neighboring pixels.

There are two fundamental morphological operations:

1. Erosion: It is used to erode the boundaries in image. It traverse every pixel one by one and check if all the pixels present that kernel is 1 or not. If all pixels are than assign 1 to the pixel at structuring element’s origin, otherwise assign 0. It is used to remove small white noise from the image.

Edge detection can also be done using erosion by subtracting the eroded image from the original image.

In case of multi-channel images, erosion is performed on each channel independently.

**cv2.erode( source, kernel, iteration )** function is used to apply erosion in OpenCV. The function works in in-place mode and has three arguments:

1. source: it signifies the source image on which transformation has to be done.
2. kernel: it specifies the kernel or structuring element.
3. iteration: We can perform erosion on the same image multiple times, hence iteration specifies number of times the operations has to be performed.

Python code:

# Importing libraries

import cv2

import numpy as np

#read input image

img = cv2.imread('lady.jpg',0);

kernel= np.ones((5,5),np.uint8); #kernel of shape (5,5) filled with all 1's

erode = cv2.erode(img,kernel,1);

diff = (img-erode).astype('uint8'); #difference of eroded image from originalimage will give us image boundaries

cv2.imshow('Original',img);

cv2.imshow('erode image',erode);

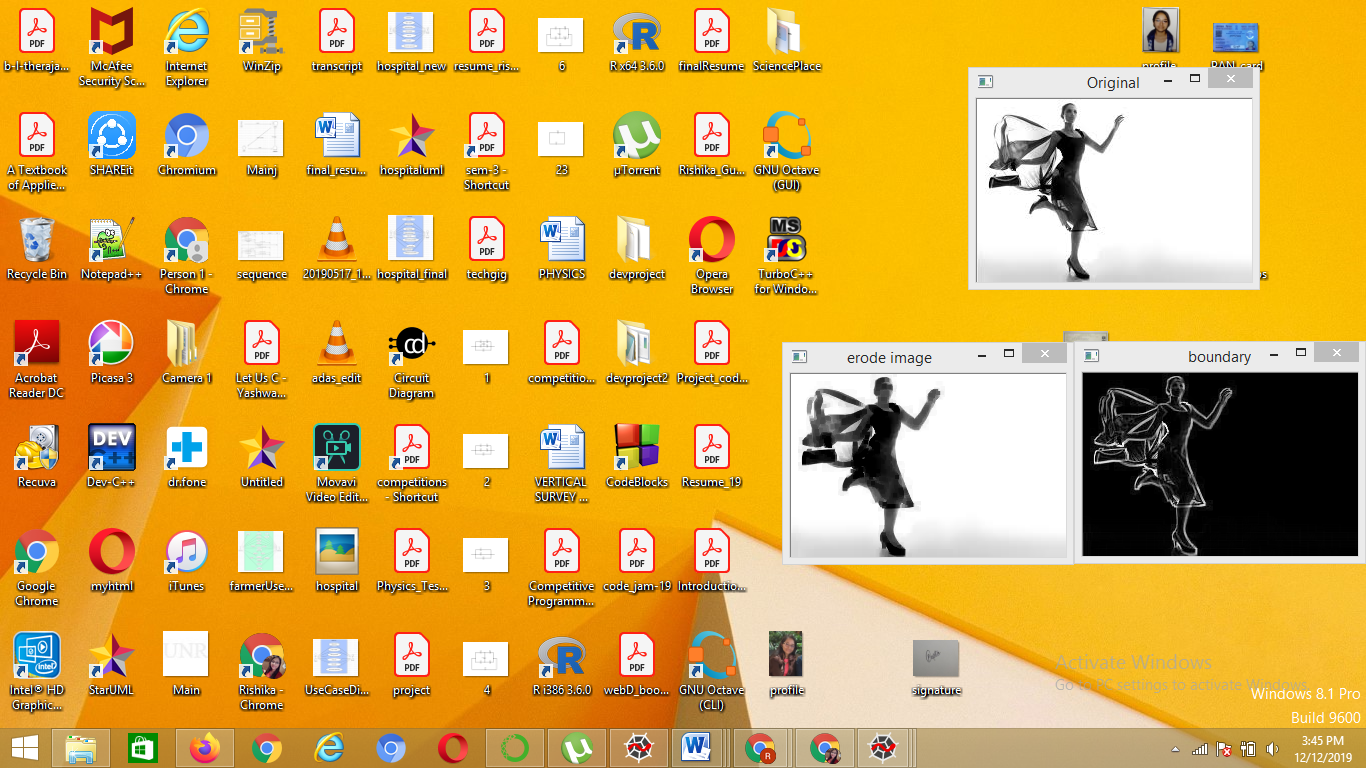
cv2.imshow('boundary', diff);

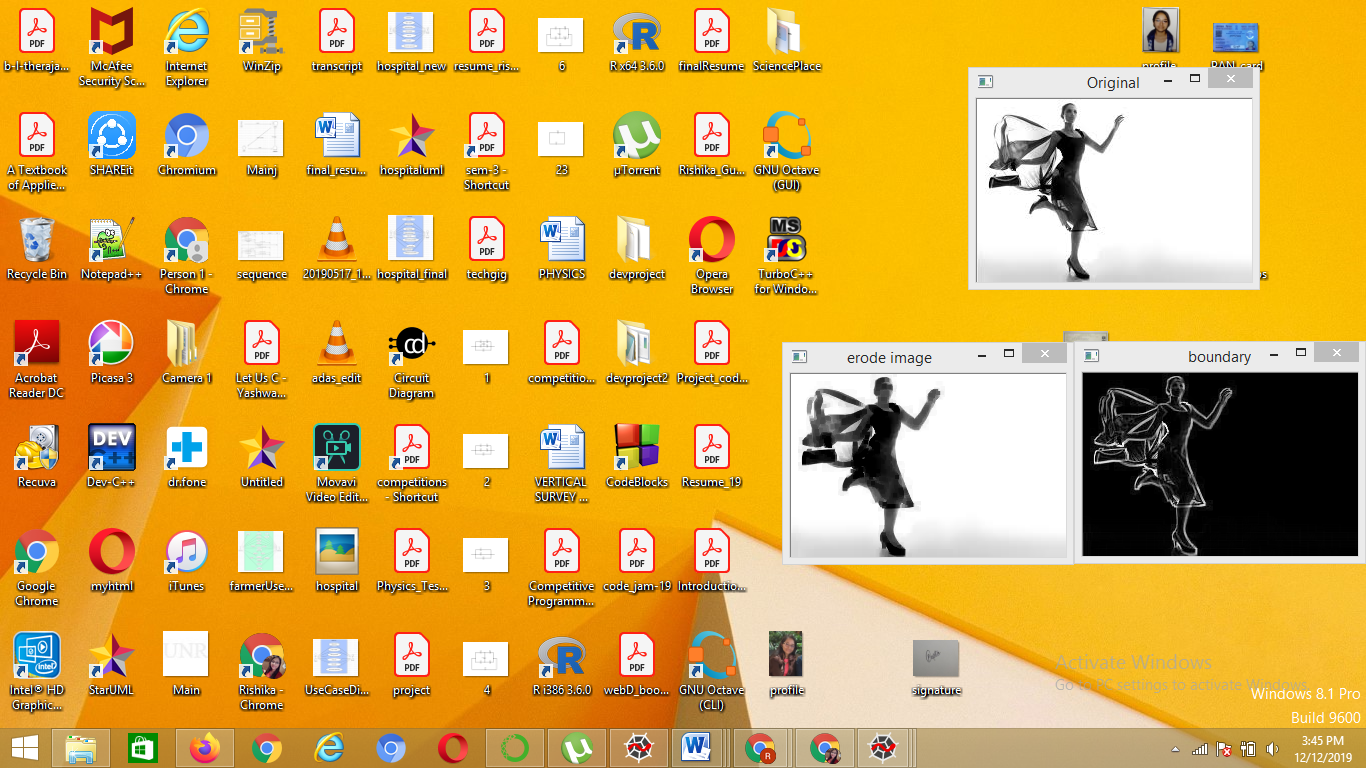
#wait for 10 sec

cv2.waitKey(10000);

cv2.destroyAllWindows();

OUTPUT:





1. Dilation: It is just the opposite of erosion and is used to add a layer to pixel to the inner and outer boundaries of the image.  Here, a pixel element is assigned 1 if atleast one pixel under the kernel not 0. So it increases the white region in the image.

**cv2.dilate(source, kernel, iteration )** function is used to apply erosion in OpenCV. The function works in in-place mode and has three arguments:

1. source: it signifies the source image on which transformation has to be done.
2. kernel: it specifies the kernel or structuring element.
3. iteration: We can perform erosion on the same image multiple times, hence iteration specifies number of times the operations has to be performed.

Python code:

# Importing libraries

import cv2

import numpy as np

#read input image

img = cv2.imread('lady.jpg',0);

kernel= np.ones((5,5),np.uint8); #kernel of shape (5,5) filled with all 1's

dilation = cv2.dilate(img,kernel,1);

cv2.imshow('Original',img);

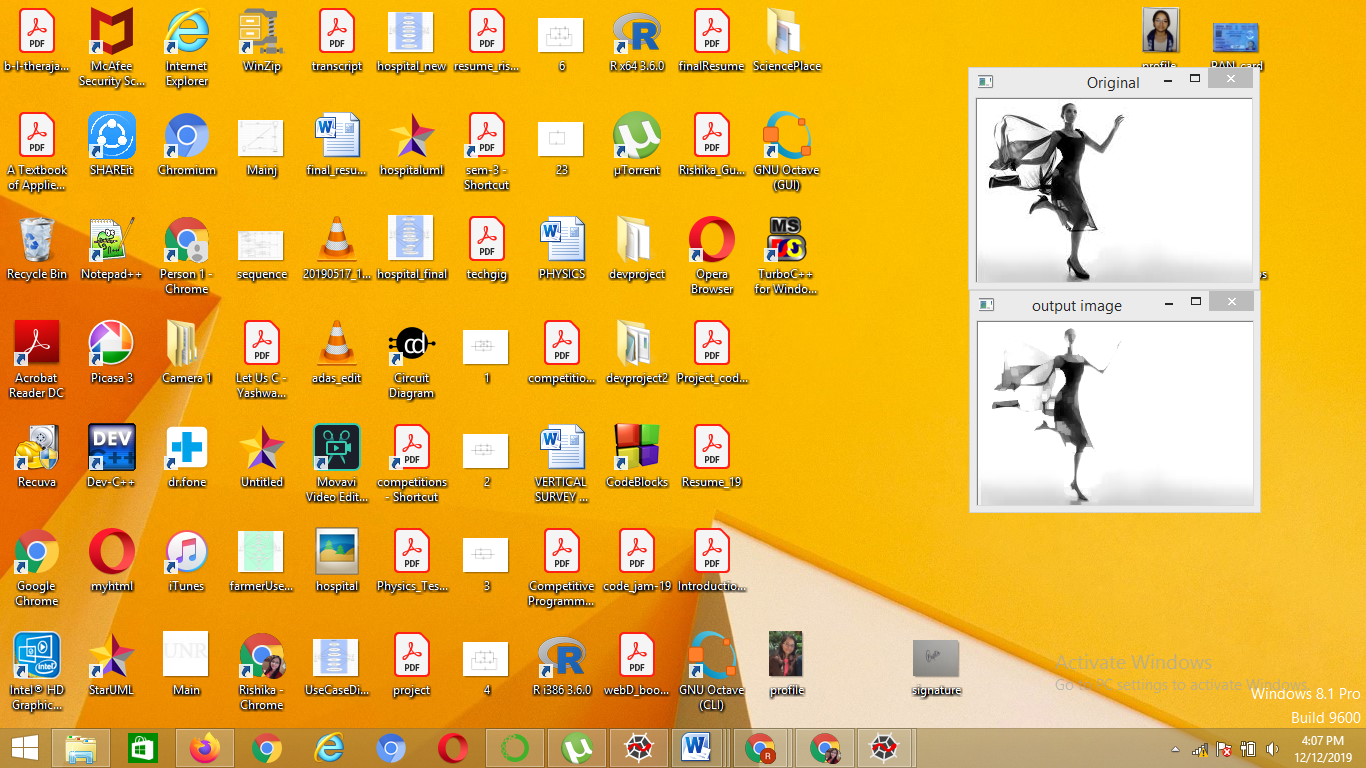
cv2.imshow('output image',dilation);

#wait for 10 sec

cv2.waitKey(10000);

cv2.destroyAllWindows();

OUTPUT:



To remove noise from the image, both erosion and dilation is used as erosion will remove the white noise but it will also shrink the boundaries of image. This shrinking can be overcome by using dilation as dilation will add inner and outer pixel layer to the boundaries.

Let’s see how it works:

Python code:

# Importing libraries

import cv2

import numpy as np

#read input image

img = cv2.imread('image.jpg',0);

kernel= np.ones((3,3),np.uint8); #kernel of shape (5,5) filled with all 1's

erode = cv2.erode(img,kernel,1);

dilation = cv2.dilate(erode,kernel,1); #perform dilation on eroded image

#to recover shrinking of image due to erosion

cv2.imshow('Original',img);

cv2.imshow('erode image',erode);

cv2.imshow('After removing noise', dilation);

#wait for 10 sec

cv2.waitKey(10000);

cv2.destroyAllWindows();

OUTPUT:

