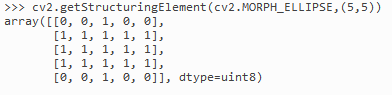
Submission of Assignment

NAME:-Adarsh Saraf Role:-Data Scientist

1. Instructions to run the code:-
   1. Make sure you have NUMPY and OPENCV installed .
   2. The code is written according to the given format that is python rice\_analysis.py ( provide path to the image you want to test).
   3. Open command prompt/anaconda prompt whichever installed and the change the directory to the folder containing the script.
   4. Run the code using :- python rice\_analysis.py ( provide path to the image you want to test) .
2. Explanation of the code:-

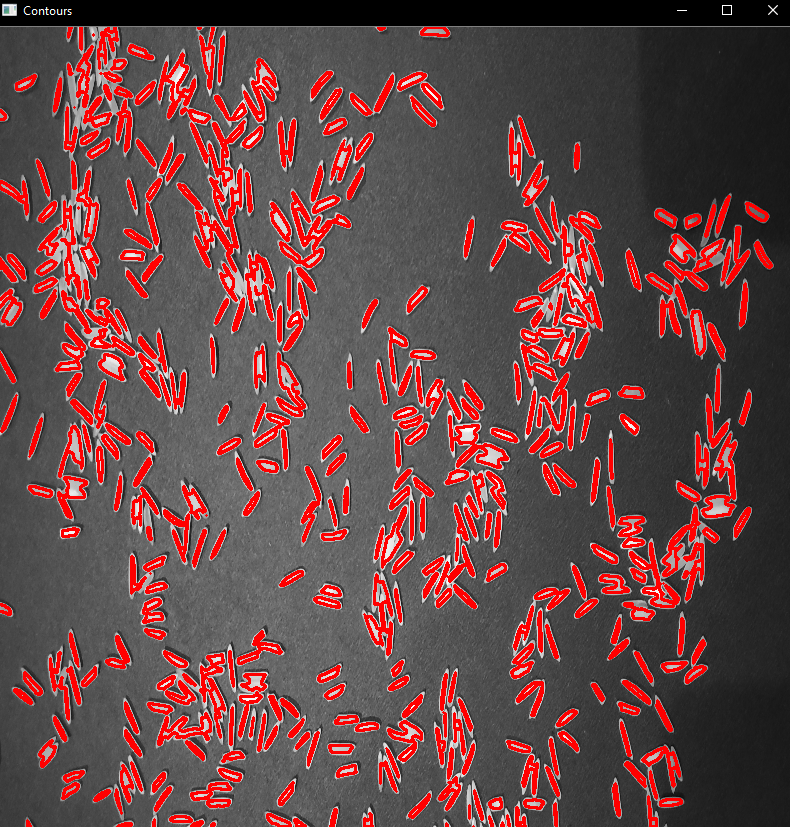
* So the first step was to read the image and resize it to a dimension of your need in this case i have (800x800).
* Then we convert the image to grayscale so we can provide that as an input to our upcoming threshold operation.
* I used adaptive threshold because , after going through the images given in the folder Rice\_images I observed a difference in the lighting conditions in certain parts of the image .eg:-broken\_grain\_1,broken\_grain\_2,broken\_grain\_3 etc have a rectangular object which is creating a difference in the lighting conditions, initially I used Otsu Thresholding but after it failing to detect the rice grains in the poor light conditions I had to use adaptive thresholding.
* After applying adaptive thresholding, I applied eroding on the thresholded image using getstructuringelement , in which I used an morph\_ellipse parameter which basically creates a elliptical shaped kernel,the reason I chose this was it gave me the best results.



* And here you will be able to see the eroded image that is formed after performing adaptive thresholding and eroding.In this example I have used mixed\_grain\_1 and we can see the output image.



* After eroding operation I applied the findcontours operation on the eroded image, and we get the number of contours detected , here we use cv2.RETR\_EXTERNAL which basically gives outer contours and I used cv2.CHAIN\_APPROX\_SIMPLE which removes all the redundant points and compresses the contour and saves memory.
* After that I find the area covered by every contour and after taking the mean of all the areas of the three broken grain images which comes to be 37, and on the basis of that if lets say a contour area is more than 37 its considered as not broken and if it falls under 37 its considered as a broken grain.
* After that I just show the number of rice grains that I have been able to detect and the percentage of broken rice grains.
* You can also see the contours formed and the rectangles of the contours .

.

LIMITATIONS:-

1. The algorithm athough can detect most of the rice grains that have some distance separating each other.It faces problem when the two grains are clustured up or joint.
2. Segregating rice grains on the basis of area is not the best solution as not ever rice grain is of the same length or the same area .
3. The are of some contours are extremely large compared to others because the rice grains are clustered up or touching each other with no space in between them .This creates false mean values and eventually false broken rice percentage value

IMPROVEMENTS:-

1. Performing more morphological operations on the image , like eroding and dilating without actually destroying the image and getting finer outputs.
2. Applying edge detection to the input image and based on the edges deciding the contours.
3. Applying better contouring methods to get more accurate results for both the results that are the number of rice grains and the percentage of broken rice grains.

**Conclusion:-**

This is my submission for the task of Computer Vision for the role of Data Scientist.