Assignment1 report

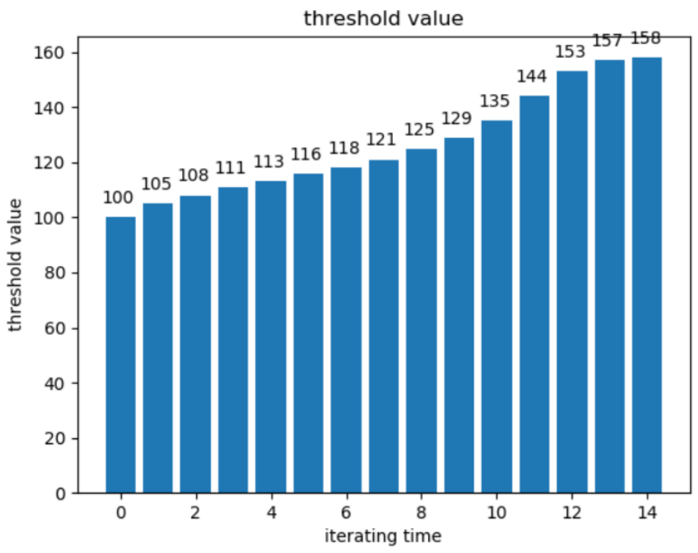
YUE HE z5243835

All the codes have been tested through vlab using python 3.7.3

Task1:

While choosing the threshold, because value of pixels are in range 0 to 255, so I decide to choose 100 to be the threshold because it is convenient and also close to 255/2.

Then we change all the pixel's value into 255 or 0. Then we can see lot's of black spots which are noise not rice, and also an image of labels will be created at the same time just as the picture below:



and this is all the work for Task1

Task2:

I write a medianblur filter on my own, and successfully wipe out the noise.

I choose the filter size to be 5X5, because 3X3 is too small and some noise still remain after filter is used.

And the pixels on the edge of the picture are changed to 255.

For a filter size 5, the edge is 2 pixels wide.

Then I follow the directions given by teacher to finish two-pass.

I create a list to record all the pixels' neighbors.

Such as:

If 2 is next to 2 , 3 is next to 2, 4 is next to 2, 5 is next to 6

The list will be like:

[[2] , [3,2], [4,2],[5,6]]

Then I change the list into [[4,3,2] , [5,6]]

The only difference is that in the second step of two-pass, in standard two-pass, if labels 2,3,4 are connected together, while label 3 is found, we should change it into 2.

In my code, I change it into the first element in its' group.

For example: from the list above, we now 4,3,2 are connected together, when we find label 2, we change it into 4, not 2, because 4 is the first element in [4,3,2]

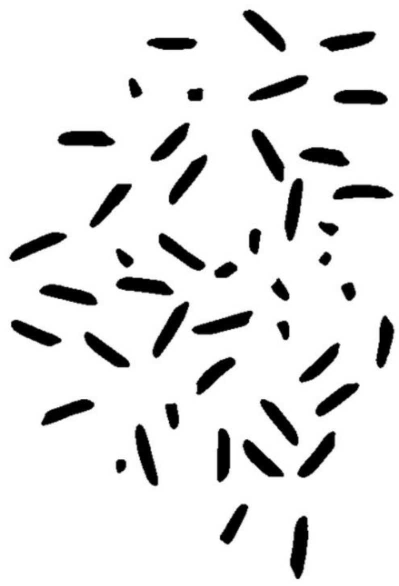
This method can save some time as we don't have to sort all the

elements in [4,3,2] to find the smallest number. But still, we can change all the labels which are connected into the same value.
So I think this is a good idea!

Task3:
To find a proper value to judge the rice is damaged or not. I count the number of pixels who belong to rice. Then number of pixels divided by rice numbers is the average pixels every rice has. Let's call this result average_size.
As we know some damaged rice are mixed inside the picture , average_size must be smaller than normal rice. However, rice are not as the same size, some rice are smaller than usual even if they are not damaged.
So I choose 2/3*average_size to be the min_area.
For image1, min_area = 421
For image2, min_area = 434
For image6,min_area = 205
For image7,min_area = 238
As you can see rice's size is changing from picture to picture, so there is no fixed min_area for all pictures.

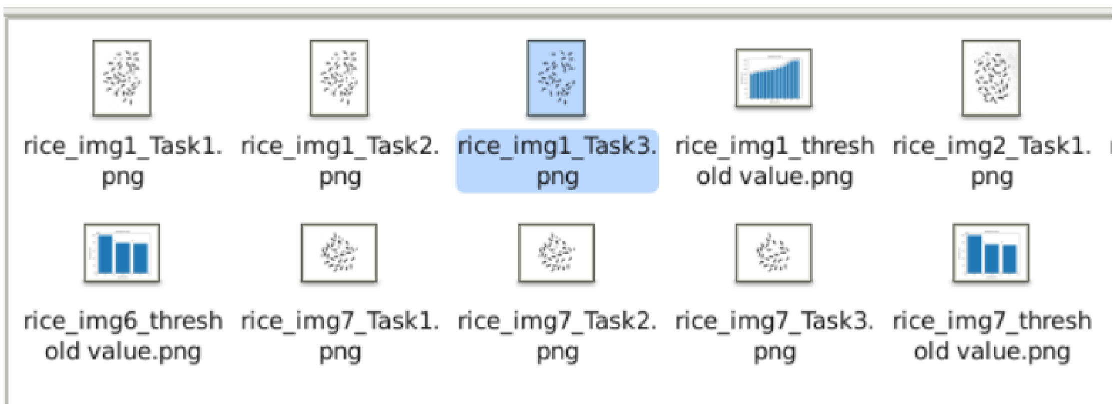
And an example is given below:

On the left side is task2's filtered image from sample image1, On the right side is task3's image from sample image1 which has ignored the damaged rice.



Of course if a wrong value is chosen, some damaged rice will remain.

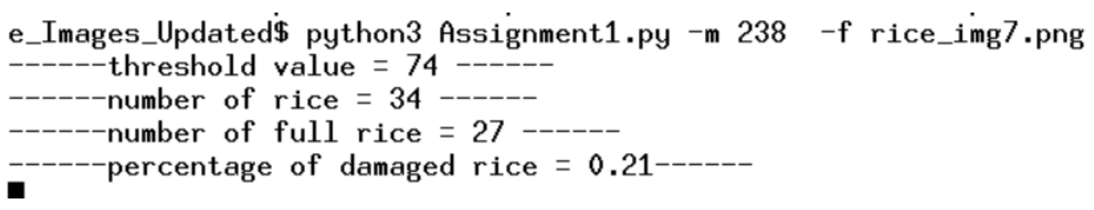
All the images(Task1.png,Task2,png,Task3.png and the threshold value image required in Task1) are stored in subfolder as Assignment1.pdf required.



rice_img1_Task1.png
rice_img1_Task2.png
rice_img1_Task3.png
rice_img1_threshold value.png
rice_img2_Task1.png
rice_img6_threshold value.png
rice_img7_Task1.png
rice_img7_Task2.png
rice_img7_Task3.png
rice_img7_threshold value.png

They can be easily recognized by their file names.

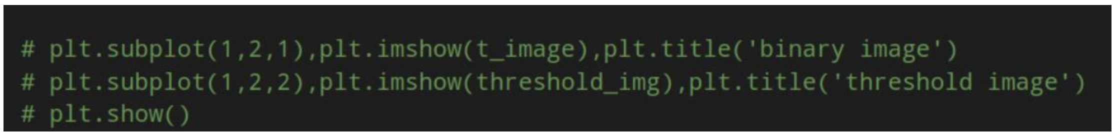
All sample images, outputs will be contained in the zip file.



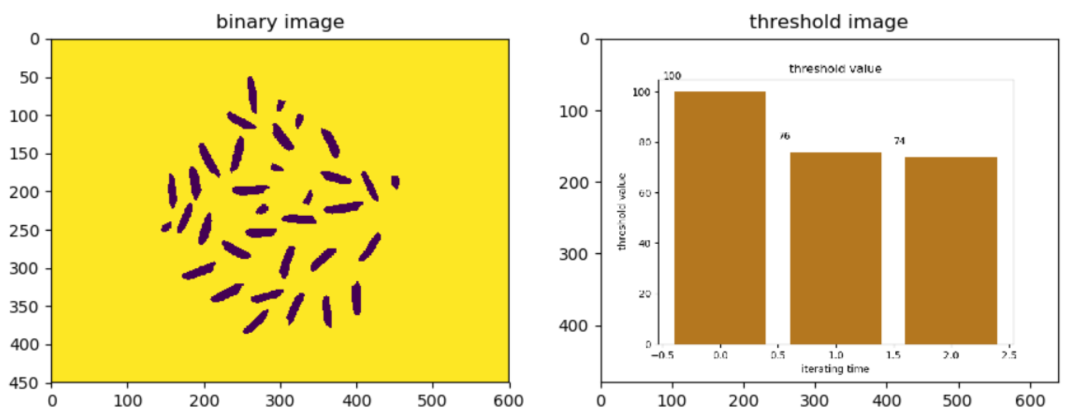
```
e_Images_Updated$ python3 Assignment1.py -m 238  -f rice_img7.png
------threshold value = 74 ------
------number of rice = 34 ------
------number of full rice = 27 ------
------percentage of damaged rice = 0.21------
```

Normally the program can run smoothly and fast. Results will show up within 10 seconds.

Although I tried to use plt.subplot function to make the output feels better, I got a very strange result so I give up.



```
# plt.subplot(1,2,1),plt.imshow(t_image),plt.title('binary image')
# plt.subplot(1,2,2),plt.imshow(threshold_img),plt.title('threshold image')
# plt.show()
```



As you can see in the picture above, colors are changed due to plt.imshow() function, and extra information are added to the image which may cause misunderstanding. So I gave up using plt.subplot() function

Above are all the works in this Assignment, thanks for your reading, and have a nice day!