CSCI677 YIYUE ZHANG

HW2 Part a

1. In this part, I implemented and tested a Mean Shift Segmentor with input images in LAB color space. I used a level-1 pyramid for this Segmentor. I also did experiments to test this Segmentor with different spatial window radius values (sp) and different color window radius values(sr).

    - '''Path of Three Sample Images and Their Corresponding Ground Truth'''
    - '''Read Three Sample Images and Their Corresponding Ground Truth In'''
    - '''Convert RGB images to LAB color space'''
    - '''Save Original LAB Images'''
    - ''' Input Parameters '''
    - ''' sp : spatial_window_radius'''
    - ''' sr : color_window_radius'''
    - '''   Experiments   '''
    - ''' sp = 30, sr = 30 '''
    - ''' sp = 10, sr = 10 '''
    - ''' sp = 60, sr = 60 '''
    - ''' sp = 10, sr = 60 '''
    - ''' sp = 60, sr = 10 ‘''
    - '''Input Image can be RGB color space or any three-channel color space'''
    - ''' Meanshift Segmentation for Image 300091'''
    - ''' Meanshift Segmentation for Image 101085'''
    - ''' Meanshift Segmentation for Image 253027'''

2. I did 5 experiments to test the functionalities of two applicable parameters—spatial window radius (sp) and different color window radius (sr). The five experiments are:

- ''' sp = 30, sr = 30 '''
- ''' sp = 10, sr = 10 '''
- ''' sp = 60, sr = 60 '''
- ''' sp = 10, sr = 60 '''
- ''' sp = 60, sr = 10 '''

These fives experiments testes the moderate and extreme (max and min) values of sp and sr in different scenarios.

I. For **first** experiment, I choose sp and sr to be moderate values, with sp = 30 and sr = 30. This will be used as a sample to compare with other experiments.

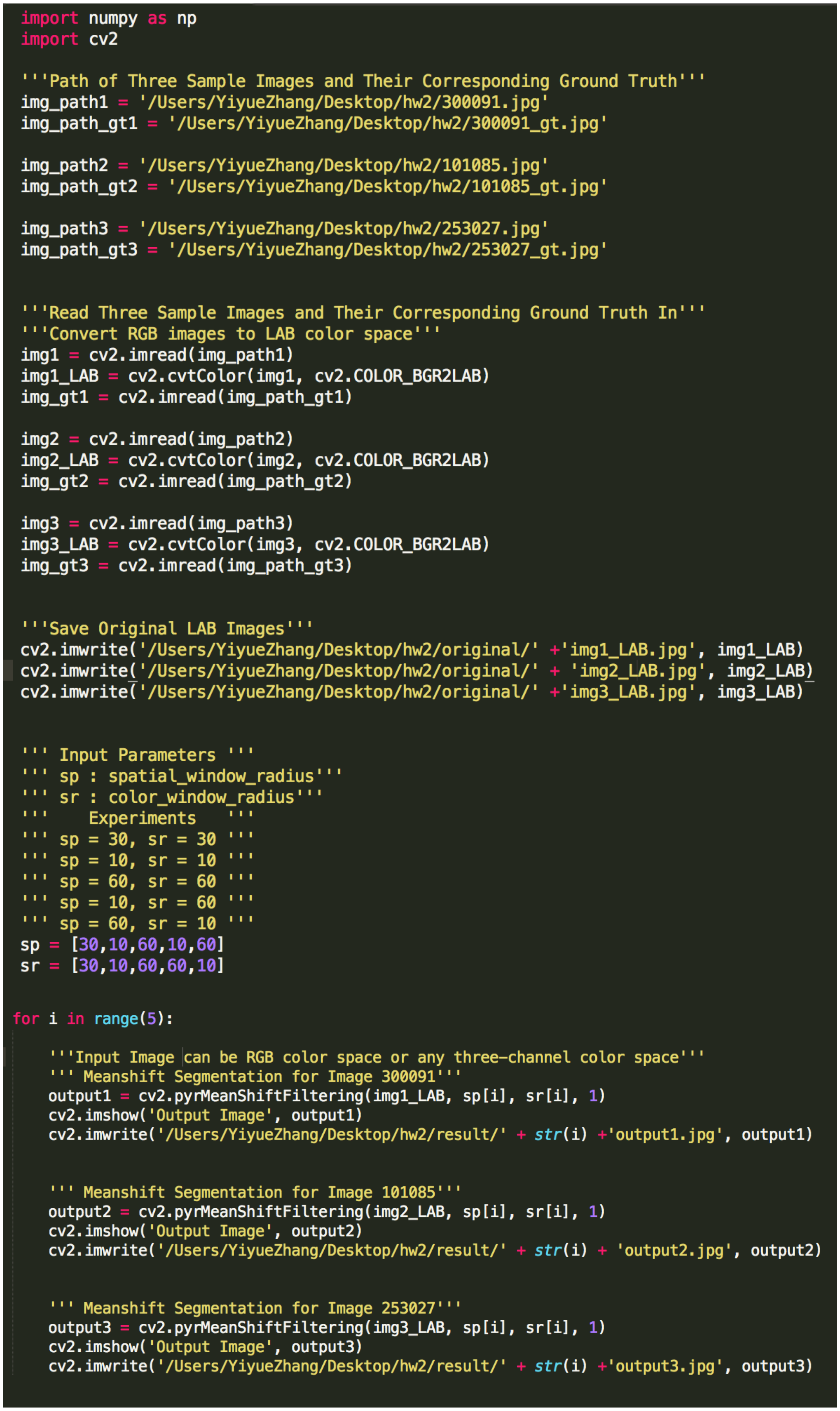
II. For **second** experiment, I choose sp and sr to be smaller values than first experiment. This is to test the influence of small sp and sr to the output.

III. For **third** experiment, I choose sp and sr to be larger values than previous experiments. This is to test the influence of large sp and sr to the output.

IV. For **fourth** experiment, I choose sp to be a small value and sr to be a larger value. This is to test the functionality of sr.

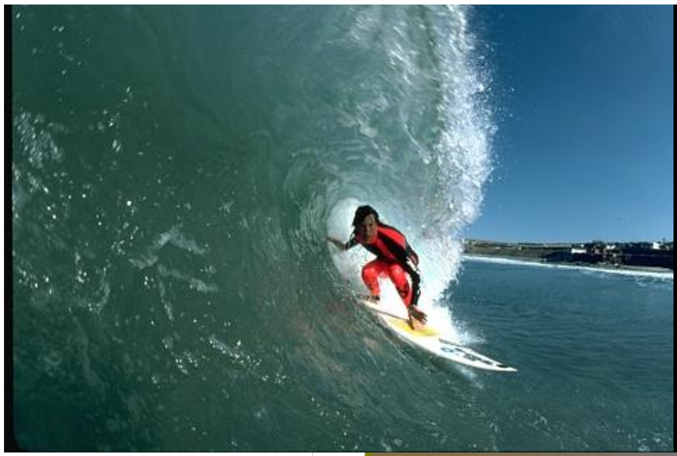
V. For **fifth** experiment, I choose sp to be a large value and sr to be a small value. This is to test the functionality of sp.

CSCI677 YIYUE ZHANG

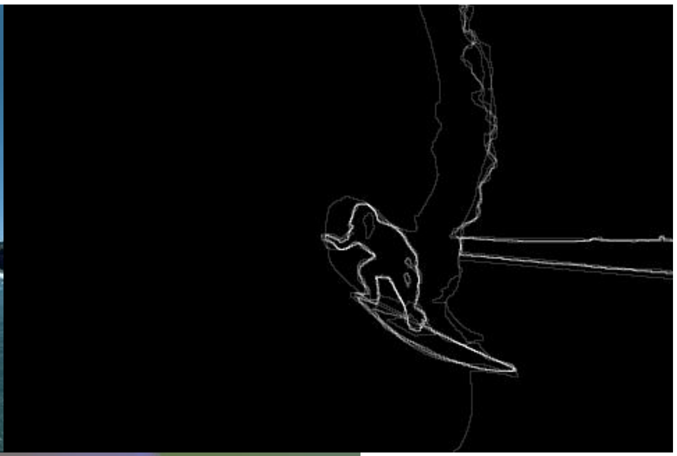


```python
import numpy as np
import cv2

'''Path of Three Sample Images and Their Corresponding Ground Truth'''
img_path1 = '/Users/YiyueZhang/Desktop/hw2/300091.jpg'
img_path_gt1 = '/Users/YiyueZhang/Desktop/hw2/300091_gt.jpg'

img_path2 = '/Users/YiyueZhang/Desktop/hw2/101085.jpg'
img_path_gt2 = '/Users/YiyueZhang/Desktop/hw2/101085_gt.jpg'

img_path3 = '/Users/YiyueZhang/Desktop/hw2/253027.jpg'
img_path_gt3 = '/Users/YiyueZhang/Desktop/hw2/253027_gt.jpg'


'''Read Three Sample Images and Their Corresponding Ground Truth In'''
'''Convert RGB images to LAB color space'''
img1 = cv2.imread(img_path1)
img1_LAB = cv2.cvtColor(img1, cv2.COLOR_BGR2LAB)
img_gt1 = cv2.imread(img_path_gt1)

img2 = cv2.imread(img_path2)
img2_LAB = cv2.cvtColor(img2, cv2.COLOR_BGR2LAB)
img_gt2 = cv2.imread(img_path_gt2)

img3 = cv2.imread(img_path3)
img3_LAB = cv2.cvtColor(img3, cv2.COLOR_BGR2LAB)
img_gt3 = cv2.imread(img_path_gt3)


'''Save Original LAB Images'''
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/original/' +'img1_LAB.jpg', img1_LAB)
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/original/' + 'img2_LAB.jpg', img2_LAB)
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/original/' +'img3_LAB.jpg', img3_LAB)


''' Input Parameters '''
''' sp : spatial_window_radius'''
''' sr : color_window_radius'''
'''    Experiments    '''
''' sp = 30, sr = 30 '''
''' sp = 10, sr = 10 '''
''' sp = 60, sr = 60 '''
''' sp = 10, sr = 60 '''
''' sp = 60, sr = 10 '''
sp = [30,10,60,10,60]
sr = [30,10,60,60,10]

for i in range(5):

    '''Input Image can be RGB color space or any three-channel color space'''
    ''' Meanshift Segmentation for Image 300091'''
    output1 = cv2.pyrMeanShiftFiltering(img1_LAB, sp[i], sr[i], 1)
    cv2.imshow('Output Image', output1)
    cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(i) +'output1.jpg', output1)


    ''' Meanshift Segmentation for Image 101085'''
    output2 = cv2.pyrMeanShiftFiltering(img2_LAB, sp[i], sr[i], 1)
    cv2.imshow('Output Image', output2)
    cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(i) + 'output2.jpg', output2)


    ''' Meanshift Segmentation for Image 253027'''
    output3 = cv2.pyrMeanShiftFiltering(img3_LAB, sp[i], sr[i], 1)
    cv2.imshow('Output Image', output3)
    cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(i) +'output3.jpg', output3)
```

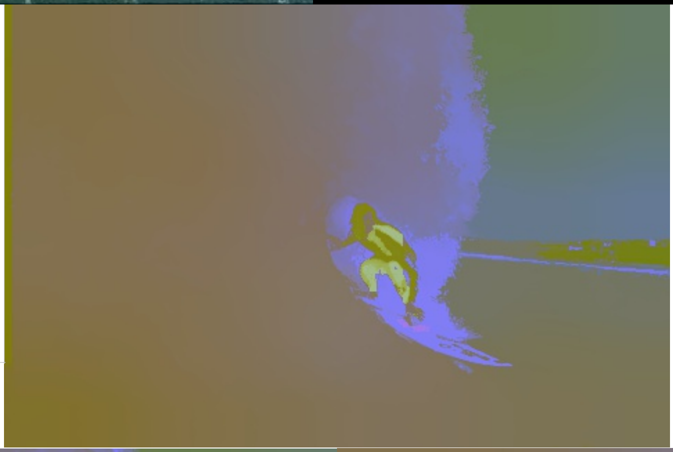
Original Image



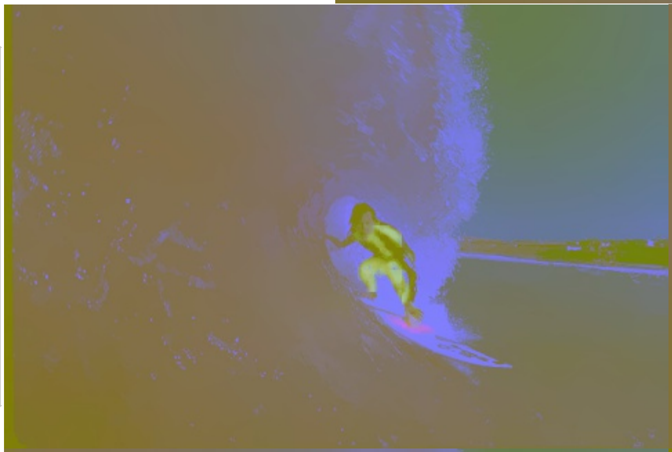
Ground Truth



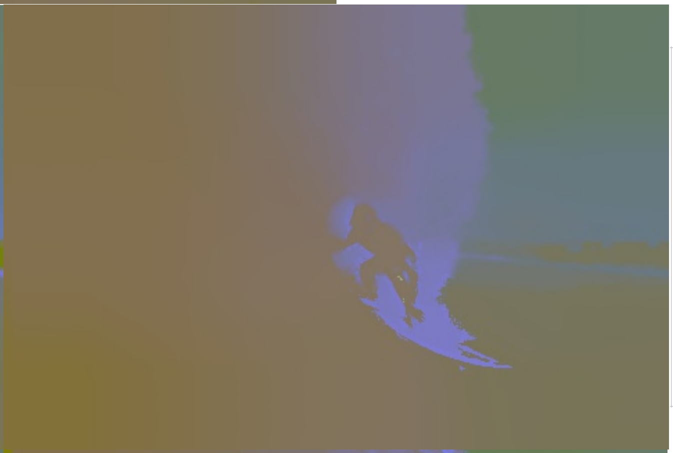
I



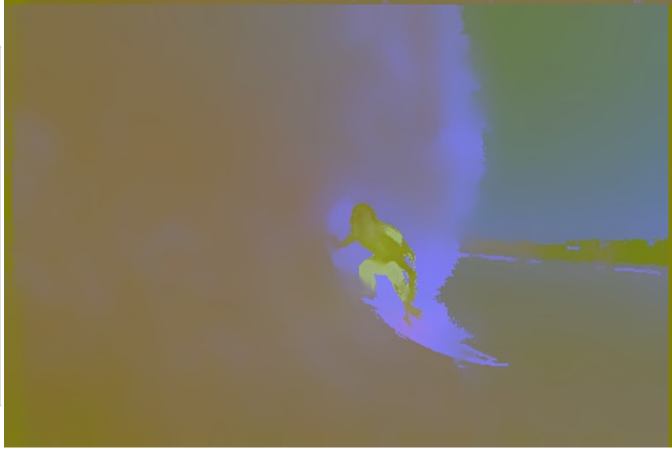
II



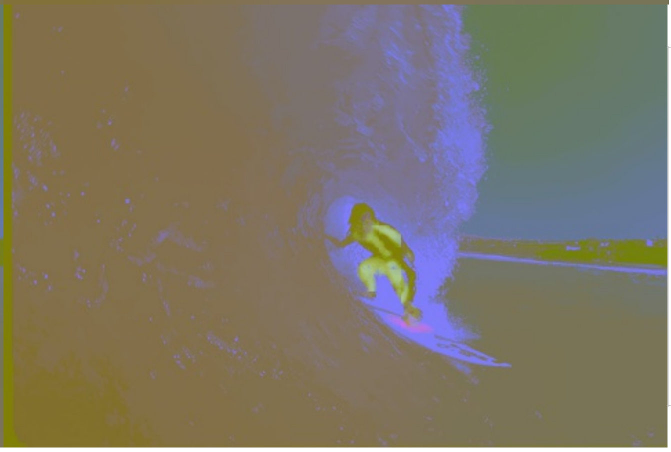
III



IV

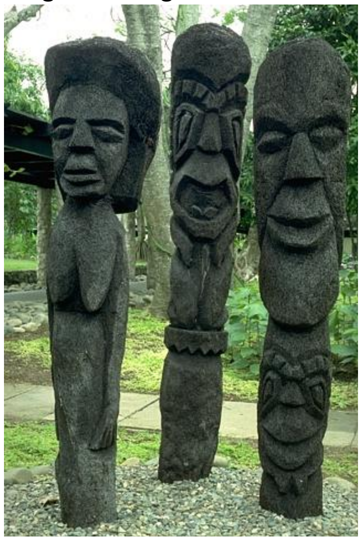


V

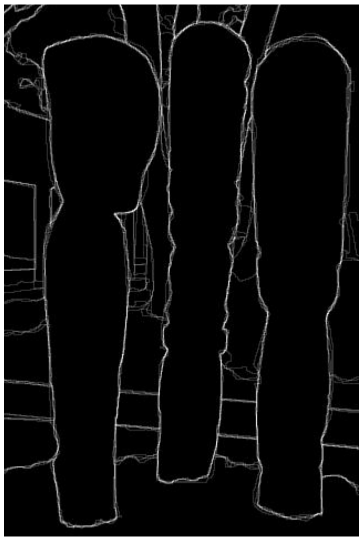


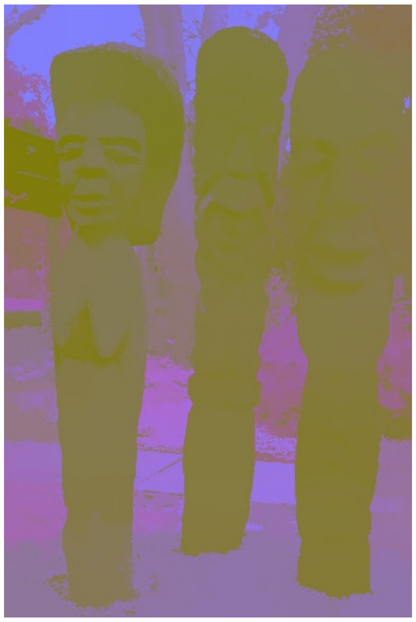
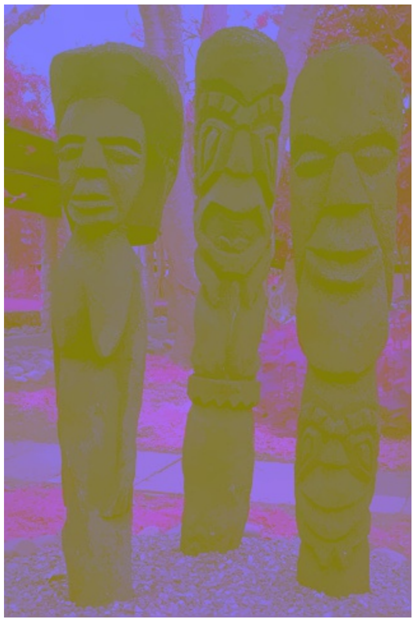
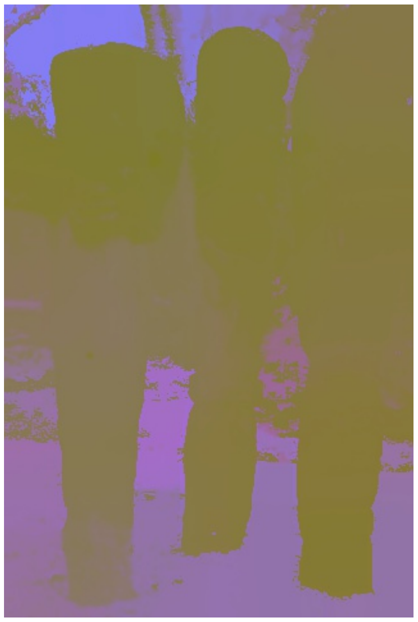
CSCI677 YIYUE ZHANG

Original Image

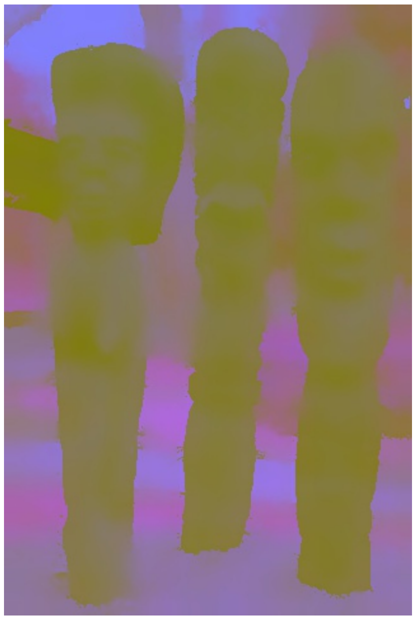
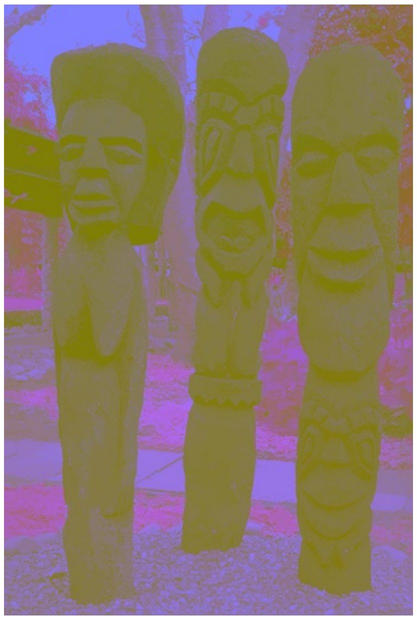


Ground Truth



I, II, III

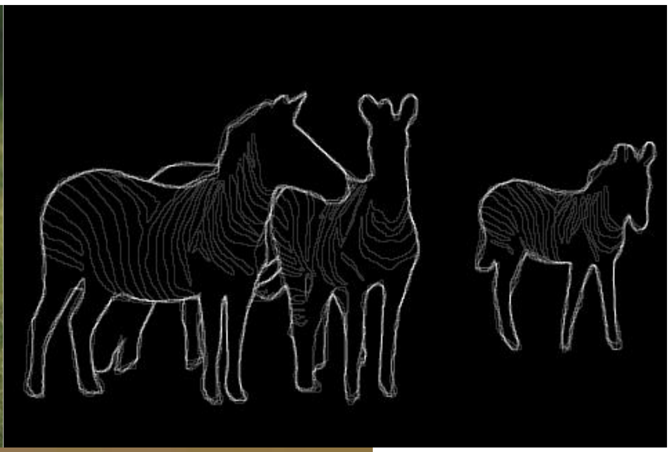



VI, V

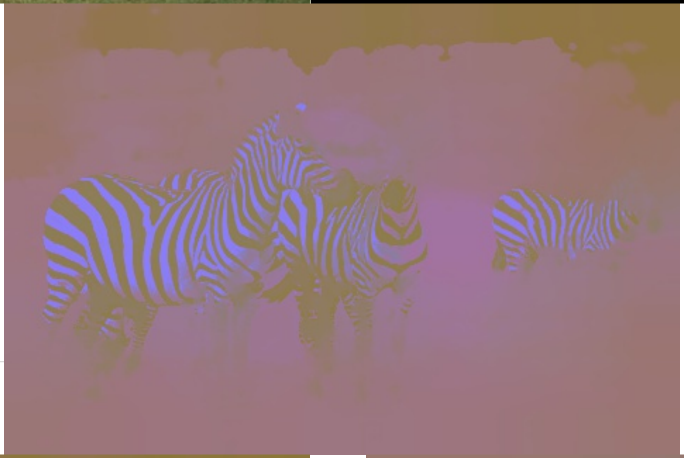
CSCI677 YIYUE ZHANG

Original Image

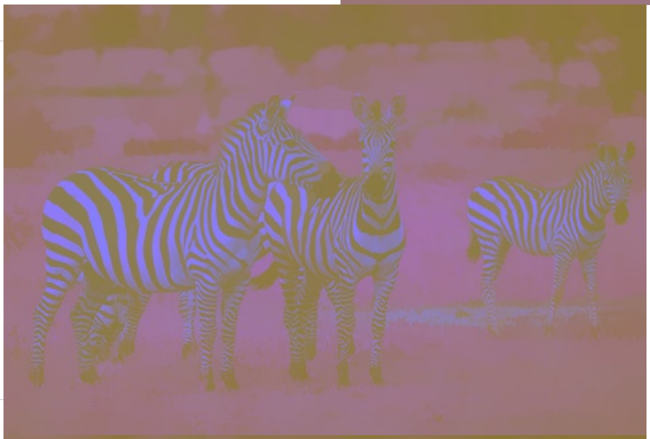
Ground Truth



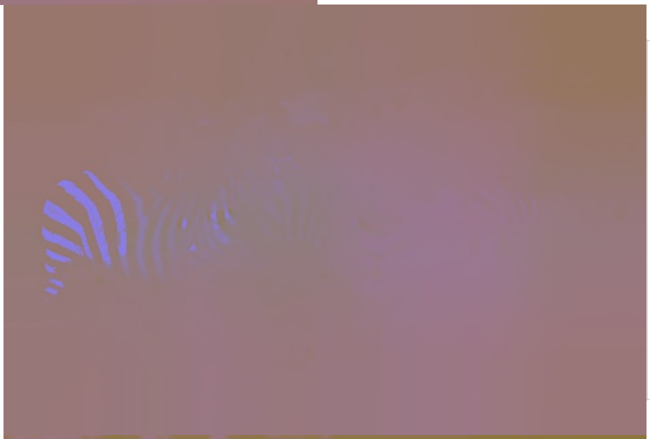
I



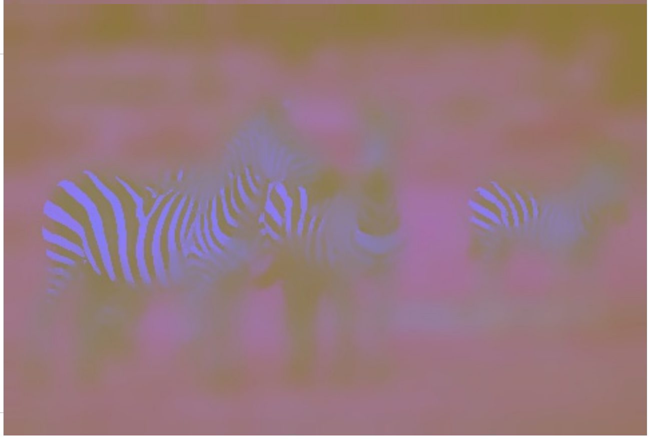
II



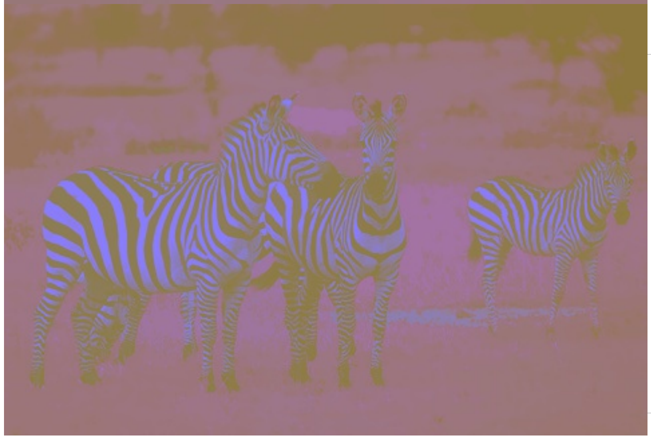
III



IV



V



3.

- ''' sp = 30, sr = 30 '''
- ''' sp = 10, sr = 10 '''
- ''' sp = 60, sr = 60 '''
- ''' sp = 10, sr = 60 '''
- ''' sp = 60, sr = 10 ‘''

In the mean shift segmentor, there are two variables—spatial windows radius and color window radius. After doing the five experiments, I observed these two parameters have different, but consistent influence to the output images.

Comparing experiment I, II, and III:

When we use large numbers for sp and sr, less boundary information will be included in the output. When we use smaller number for sp and sr, much detailed boundary information will be included in the output. This shows if we choose both sp and sr to be small, irrelevant (un-wanted) and background information will be included in the output. If we choose both sp and sr to be large, not only background, but also boundary information will be blurred out.

Comparing experiment I, IV, V:

When we set sp to be small and sr to be large, it will blur the boundary. This is because with a large sr value, information for boundary with a lower color contrast will not be shown in the output. From this, I conclude that with a large color window radius, it will remove the low contrast background feature.

When we set sr to be small and sp to be large, more detailed information in the images survived. This is because with a large sp value, information for boundary with a smaller spatial support will survive. From this, I conclude that with a large spatial window radius, small spatial support feature will survive.

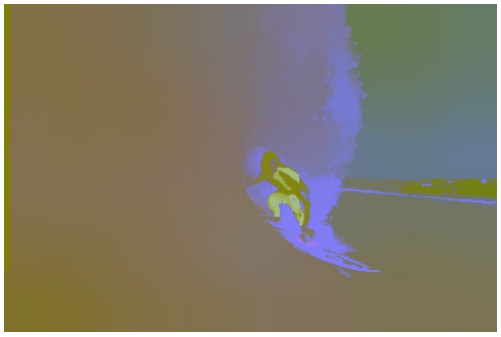
Therefore:

For first image, we want to keep some background information and the water splashes. Thus a moderate sp and sr should be used. sp = 30 and sr = 30.

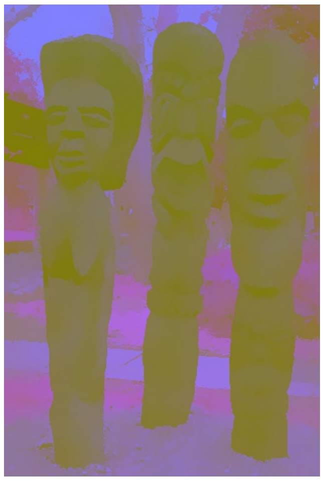
For second image, we want to remove faces from sculptures. Thus a small sp number should be used. sp = 10 and sr 30.

For third image, we want to keep feature in the skin of zebra. We should keep the color contract. Some a large sr should be used. sp = 30 and sr = 60.

sp = 30 sr = 30



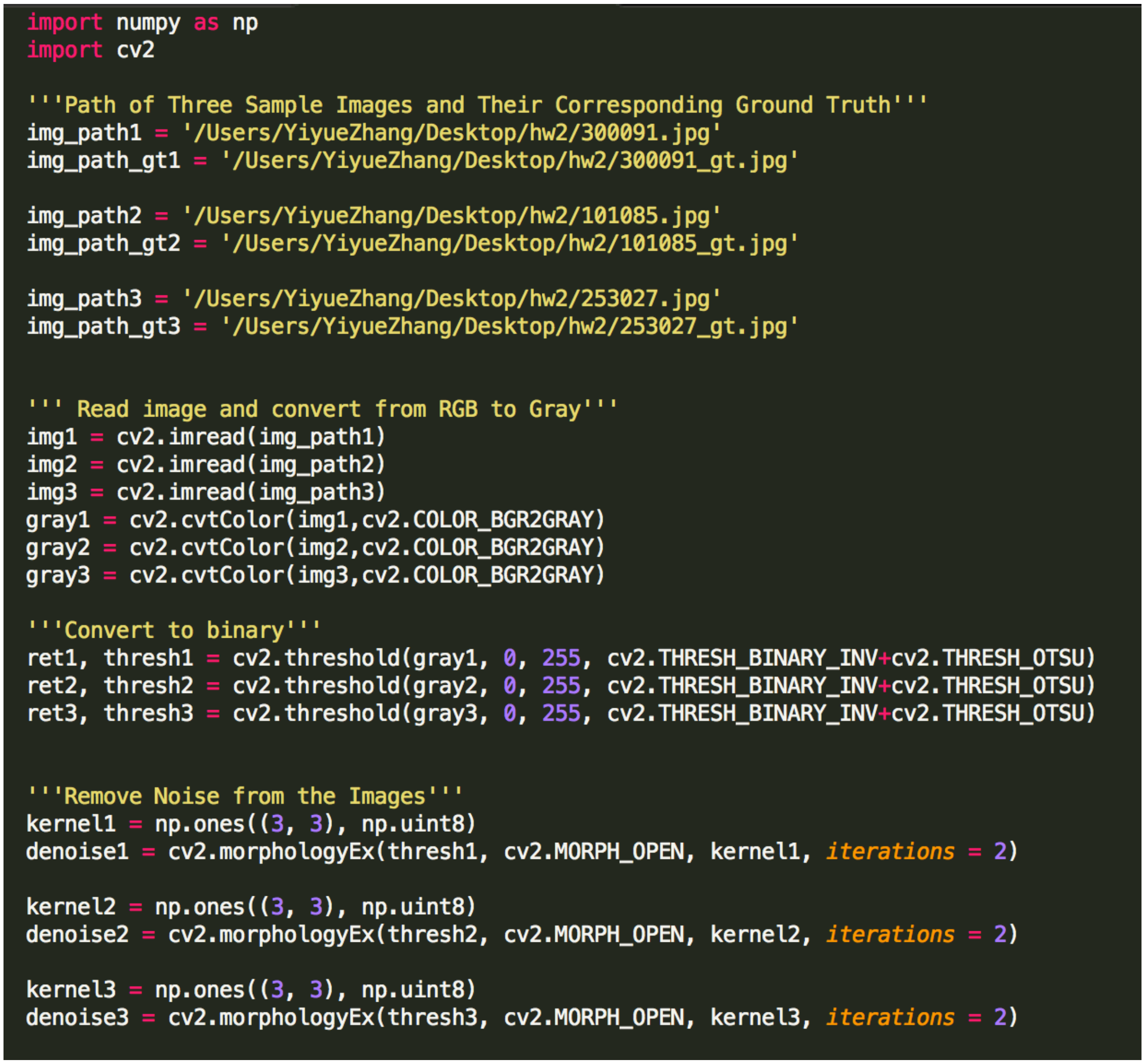
sp =10 sr = 30



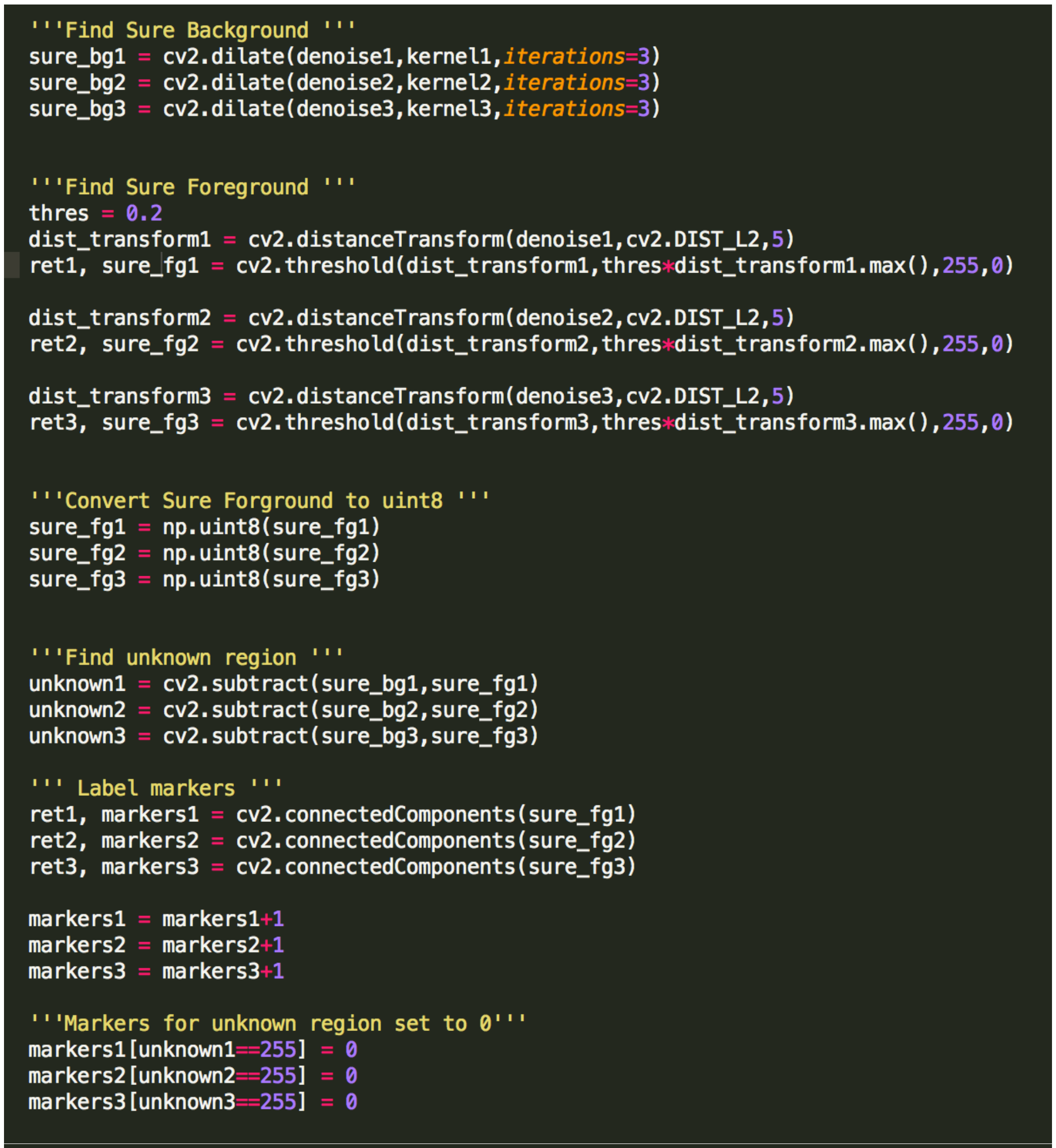
sp = 30 sr = 10

HW2 Part b

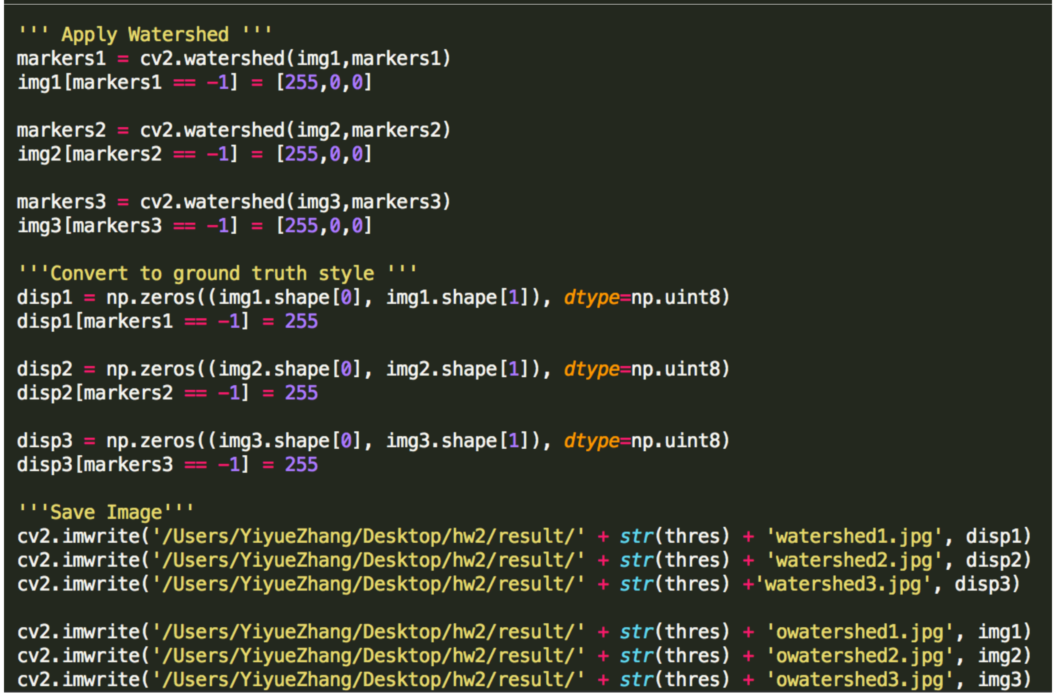
1. In this part, I implanted and tested a marker-based segmentation using Watershed Segmentor by self-defining the seeds (markers.)
   - '''Path of Three Sample Images and Their Corresponding Ground Truth'''
   - ''' Read image and convert from RGB to Gray'''
   - '''Convert to binary'''
   - '''Remove Noise from the Images'''
   - '''Find Sure Background ‘''
   - '''Find Sure Foreground ‘''
   - '''Convert Sure Forground to uint8 ‘''
   - '''Find unknown region ‘''
   - ''' Label markers ‘''
   - '''Markers for unknown region set to 0'''
   - ''' Apply Watershed ‘''
   - '''Convert to ground truth style ‘''
   - '''Save Image'''



```python
import numpy as np
import cv2

'''Path of Three Sample Images and Their Corresponding Ground Truth'''
img_path1 = '/Users/YiyueZhang/Desktop/hw2/300091.jpg'
img_path_gt1 = '/Users/YiyueZhang/Desktop/hw2/300091_gt.jpg'

img_path2 = '/Users/YiyueZhang/Desktop/hw2/101085.jpg'
img_path_gt2 = '/Users/YiyueZhang/Desktop/hw2/101085_gt.jpg'

img_path3 = '/Users/YiyueZhang/Desktop/hw2/253027.jpg'
img_path_gt3 = '/Users/YiyueZhang/Desktop/hw2/253027_gt.jpg'


''' Read image and convert from RGB to Gray'''
img1 = cv2.imread(img_path1)
img2 = cv2.imread(img_path2)
img3 = cv2.imread(img_path3)
gray1 = cv2.cvtColor(img1,cv2.COLOR_BGR2GRAY)
gray2 = cv2.cvtColor(img2,cv2.COLOR_BGR2GRAY)
gray3 = cv2.cvtColor(img3,cv2.COLOR_BGR2GRAY)

'''Convert to binary'''
ret1, thresh1 = cv2.threshold(gray1, 0, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
ret2, thresh2 = cv2.threshold(gray2, 0, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
ret3, thresh3 = cv2.threshold(gray3, 0, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)


'''Remove Noise from the Images'''
kernel1 = np.ones((3, 3), np.uint8)
denoise1 = cv2.morphologyEx(thresh1, cv2.MORPH_OPEN, kernel1, iterations = 2)

kernel2 = np.ones((3, 3), np.uint8)
denoise2 = cv2.morphologyEx(thresh2, cv2.MORPH_OPEN, kernel2, iterations = 2)

kernel3 = np.ones((3, 3), np.uint8)
denoise3 = cv2.morphologyEx(thresh3, cv2.MORPH_OPEN, kernel3, iterations = 2)
```



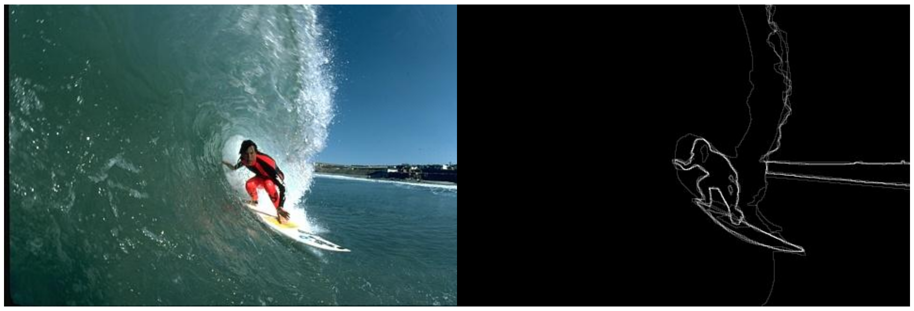
```
'''Find Sure Background '''
sure_bg1 = cv2.dilate(denoise1,kernel1,iterations=3)
sure_bg2 = cv2.dilate(denoise2,kernel2,iterations=3)
sure_bg3 = cv2.dilate(denoise3,kernel3,iterations=3)


'''Find Sure Foreground '''
thres = 0.2
dist_transform1 = cv2.distanceTransform(denoise1,cv2.DIST_L2,5)
ret1, sure_fg1 = cv2.threshold(dist_transform1,thres*dist_transform1.max(),255,0)

dist_transform2 = cv2.distanceTransform(denoise2,cv2.DIST_L2,5)
ret2, sure_fg2 = cv2.threshold(dist_transform2,thres*dist_transform2.max(),255,0)

dist_transform3 = cv2.distanceTransform(denoise3,cv2.DIST_L2,5)
ret3, sure_fg3 = cv2.threshold(dist_transform3,thres*dist_transform3.max(),255,0)


'''Convert Sure Forground to uint8 '''
sure_fg1 = np.uint8(sure_fg1)
sure_fg2 = np.uint8(sure_fg2)
sure_fg3 = np.uint8(sure_fg3)


'''Find unknown region '''
unknown1 = cv2.subtract(sure_bg1,sure_fg1)
unknown2 = cv2.subtract(sure_bg2,sure_fg2)
unknown3 = cv2.subtract(sure_bg3,sure_fg3)

''' Label markers '''
ret1, markers1 = cv2.connectedComponents(sure_fg1)
ret2, markers2 = cv2.connectedComponents(sure_fg2)
ret3, markers3 = cv2.connectedComponents(sure_fg3)

markers1 = markers1+1
markers2 = markers2+1
markers3 = markers3+1

'''Markers for unknown region set to 0'''
markers1[unknown1==255] = 0
markers2[unknown2==255] = 0
markers3[unknown3==255] = 0
```

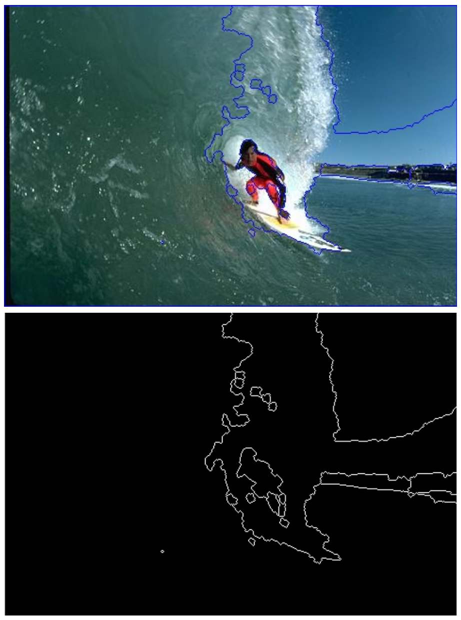


```
''' Apply Watershed '''
markers1 = cv2.watershed(img1,markers1)
img1[markers1 == -1] = [255,0,0]

markers2 = cv2.watershed(img2,markers2)
img2[markers2 == -1] = [255,0,0]

markers3 = cv2.watershed(img3,markers3)
img3[markers3 == -1] = [255,0,0]

'''Convert to ground truth style '''
disp1 = np.zeros((img1.shape[0], img1.shape[1]), dtype=np.uint8)
disp1[markers1 == -1] = 255

disp2 = np.zeros((img2.shape[0], img2.shape[1]), dtype=np.uint8)
disp2[markers2 == -1] = 255

disp3 = np.zeros((img3.shape[0], img3.shape[1]), dtype=np.uint8)
disp3[markers3 == -1] = 255

'''Save Image'''
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(thres) + 'watershed1.jpg', disp1)
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(thres) + 'watershed2.jpg', disp2)
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(thres) +'watershed3.jpg', disp3)

cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(thres) + 'owatershed1.jpg', img1)
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(thres) + 'owatershed2.jpg', img2)
cv2.imwrite('/Users/YiyueZhang/Desktop/hw2/result/' + str(thres) + 'owatershed3.jpg', img3)
```

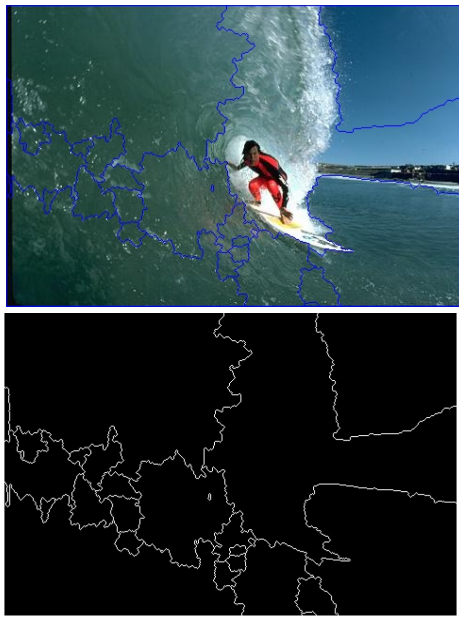
2. In this problem, there are one variable to control the functionality of Watershed Segmentor. It is the threshold for choosing the sure foreground. In this part, I did six experiments by choosing the threshold to be 0.0, 0.1, 0.2, 0.3, 0.6, 0.9.

Original Image Ground Truth

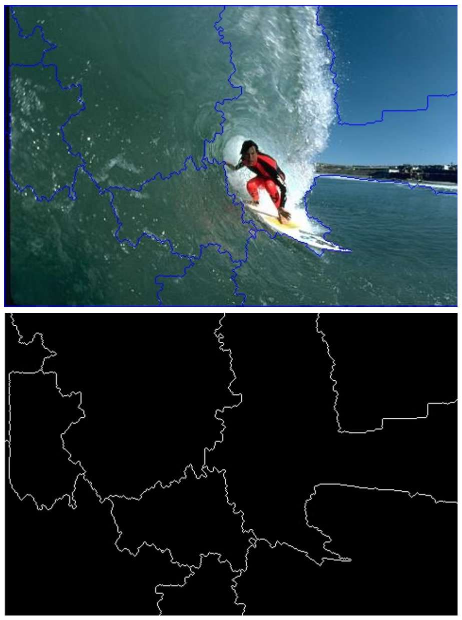




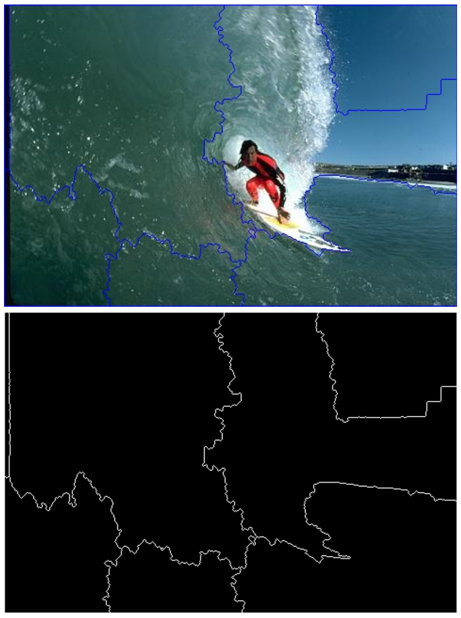
threshold = 0.0



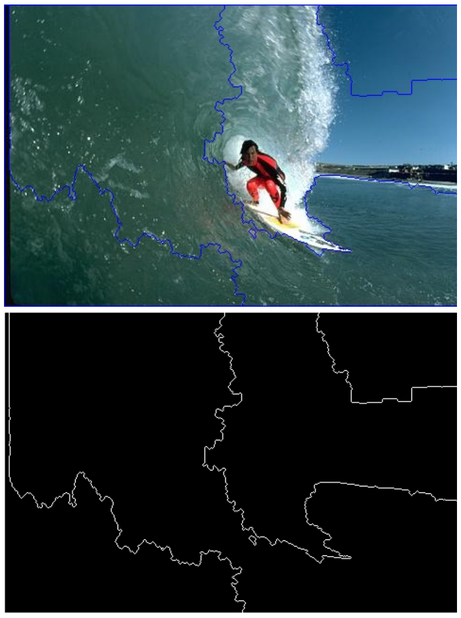
threshold = 0.1



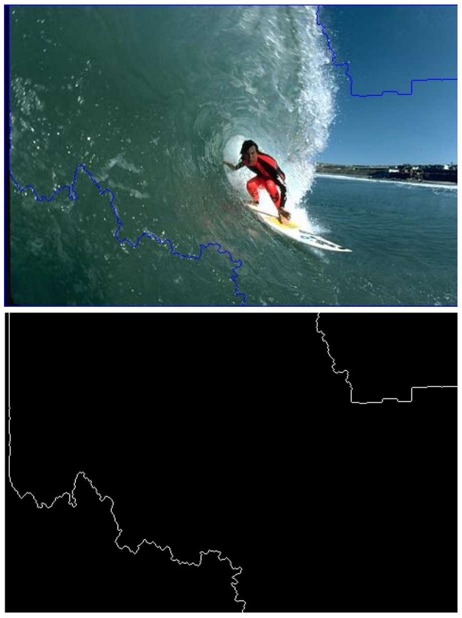
threshold = 0.2



threshold = 0.3

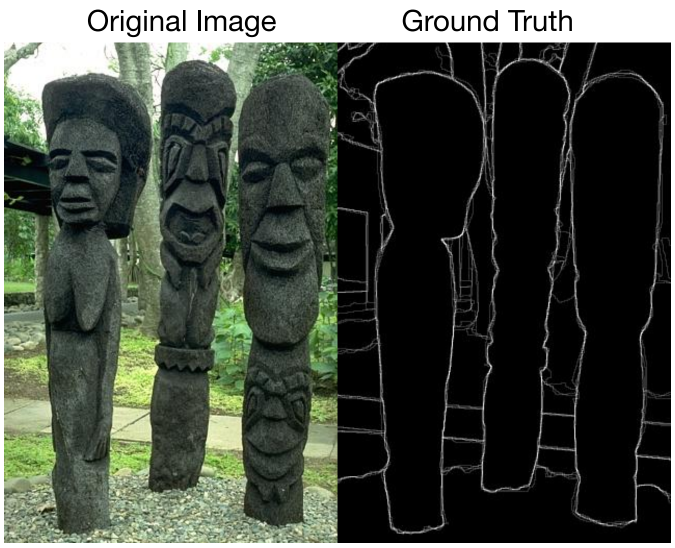


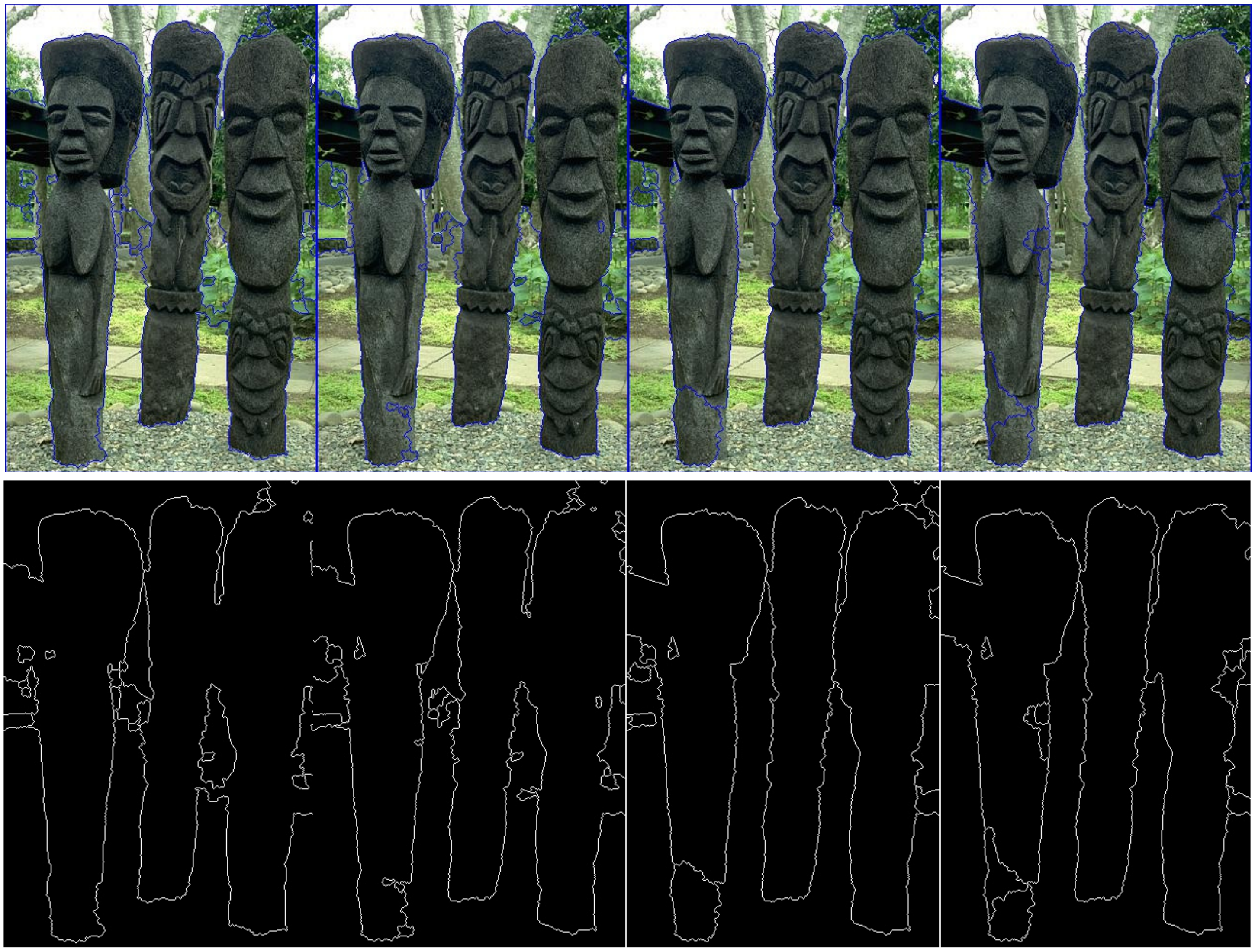
threshold = 0.6



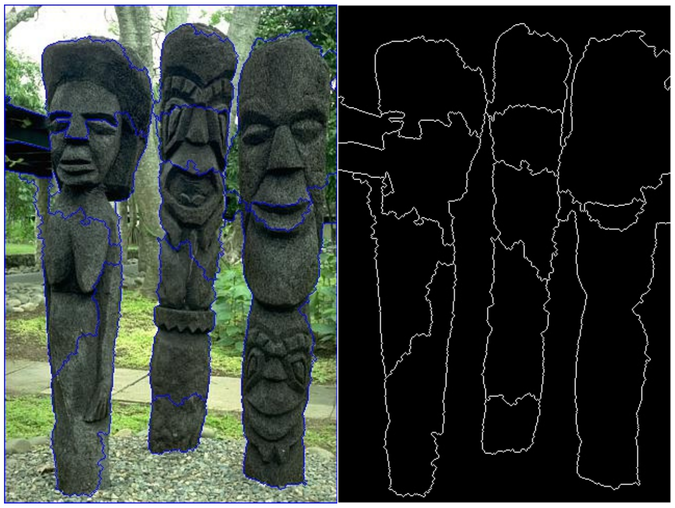
threshold = 0.9

CSCI677 YIYUE ZHANG

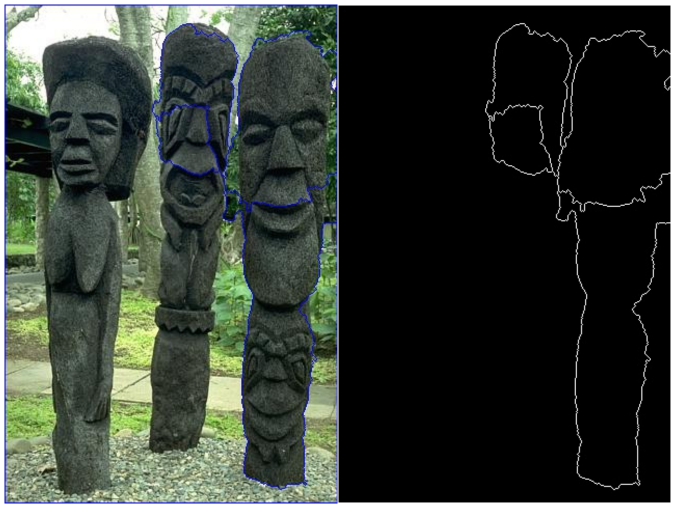




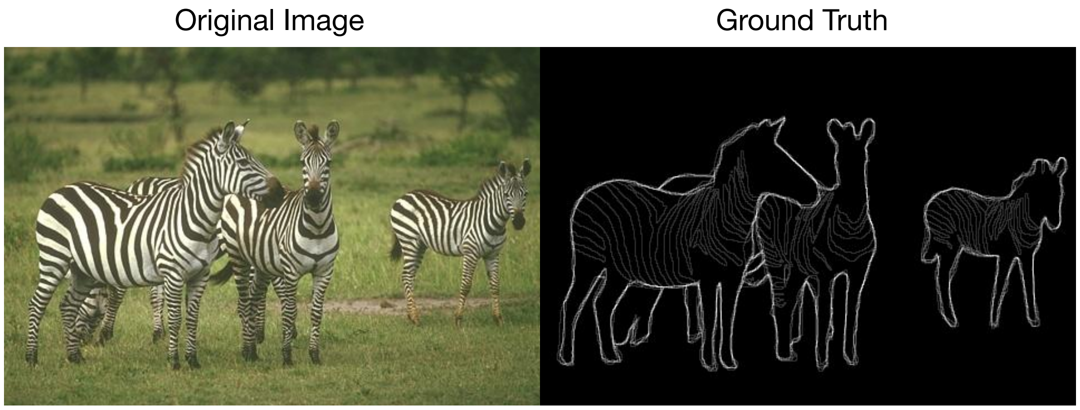
threshold = 0.0 **threshold = 0.1** **threshold = 0.2** **threshold = 0.3**

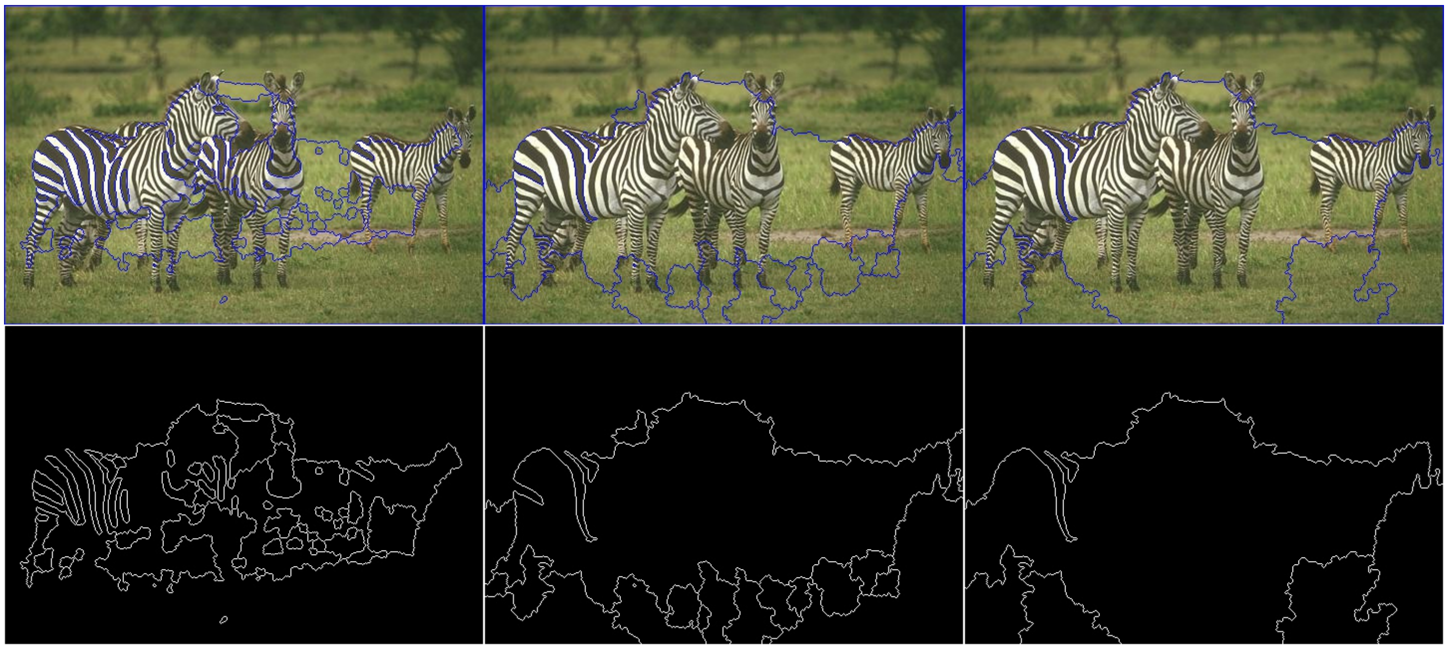


threshold = 0.6

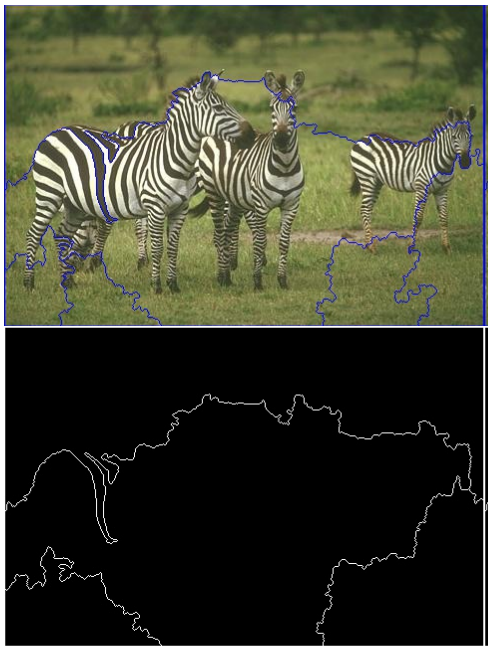


threshold = 0.9

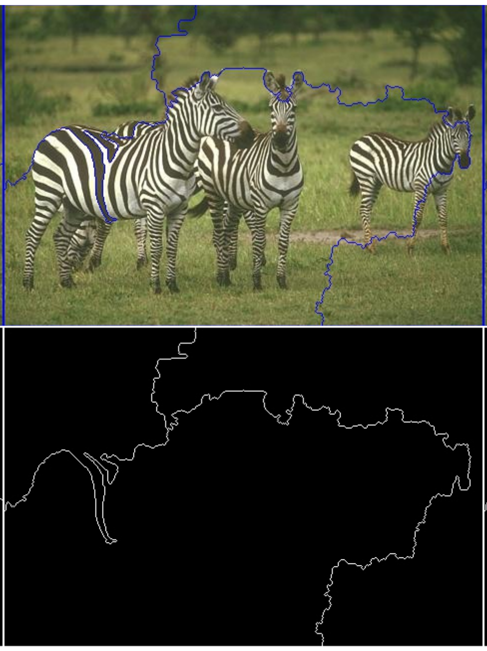




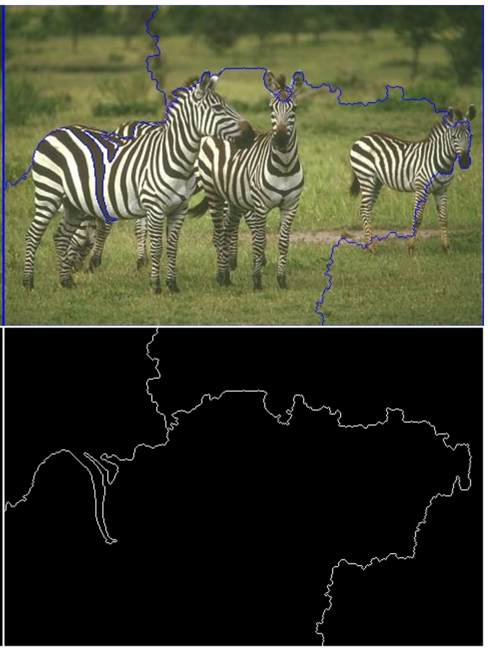
threshold = 0.0 **threshold = 0.1** **threshold = 0.2**



threshold = 0.3



threshold = 0.6



threshold = 0.9

3. For Watershed Segmentor, I used different threshold for choosing the sure foreground of image. They are 0.0, 0.1, 0.2, 0.3, 0.6, 0.9. It shows with a smaller threshold, it segments the image in a much detailed way and with a lot of un-wanted boundaries included. But, with a larger threshold, it segments the image with less details and with wanted boundary information removed. We need to choose a moderate threshold for different images. I conclude with a smaller threshold, the image is over segmented. With a larger threshold, the image is under segmented.

For the first image, when threshold equals to 0.0, it gives the best result comparing with others. It is most close to the gourd truth. For second image, when threshold equals to 0.2, it gives the best result comparing to ground truth. For third image, none of the thresholds give a good segmentation.