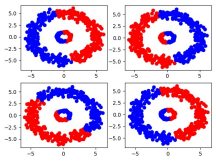
Image Processing: Homework 3

Due on December 22, 2019 at 10:00pm

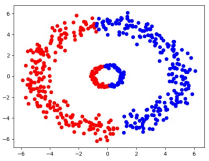
Yamin Sarcheshmehpour
94109827

Problem 1

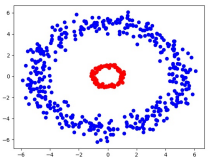
Solution The purpose of this problem is segmenting the points in *Points.txt*. At first I have used $Kmeans$, as you can see it didn't work well, I have tried it 4 times.



After that I have used $Mean\ Shift$ method but it was bad too.



By changing the coordinate space to ρ and θ and using $Kmeans$, it works well.



Describe functions

1. *k_means*: This function clusters the points using $Kmeans$ algorithm.
2. *mean_shift*: it clusters the points using $meanshift$ algorithm.
3. *change_space*: it changes the space of points from x and y to ρ and θ.

The code is in *Q1.py*, for runing the code please run *python Q1.py*.

Problem 2

Solution At this problem I tried to segment the following image using $Meanshift$ algorithms.

The result is as follows.

Describe functions

1. I have used *sklearn* package.

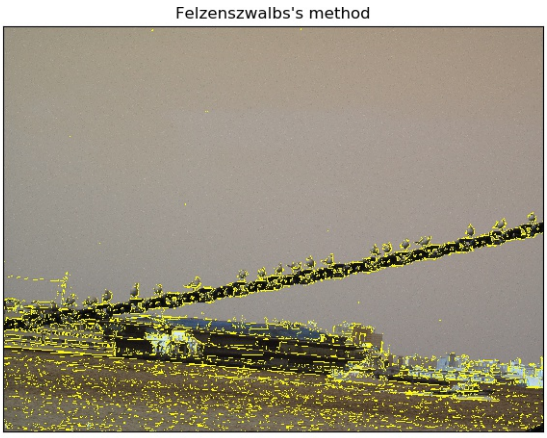
I have used *sklearn* package.
The code is in $Q2.py$, for runing the code please run *python* $Q2.py$.

Problem 3

Problem 4

At this problem I have tried to segment the following image by $Felzenswalb - Huttenlocher$ method.

The purpose was finding the birds, The result is as follows. (the result has saved with the name of $im08.jpg$ at the directory)



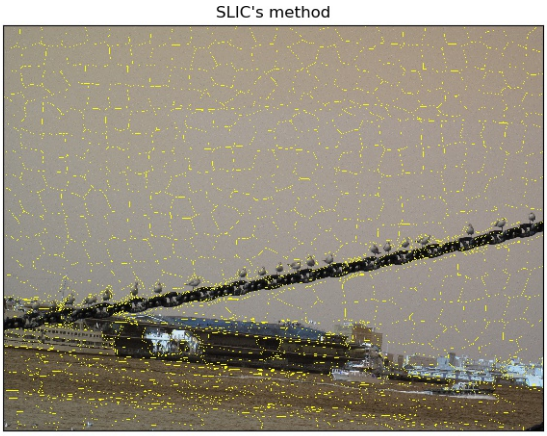
Describe functions

1. I have used *skimage* package.

The code is in $Q4.py$, for runing the code please run *python* $Q4.py$.

Problem 5

At this problem, I have tried to segment the previous image by $SLIC$ method.
The purpose was finding the birds, The result is as follows. (the result has saved with the name of $im09.jpg$ at the directory)



As you can see the $Felzenswalb - Huttenlocher$ method is better than $SLIC$ for this Image.
Describe functions

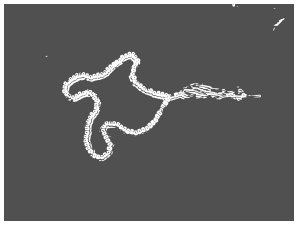
1. I have used *skimage* package.

The code is in $Q5.py$, for runing the code please run *python* $Q5.py$.

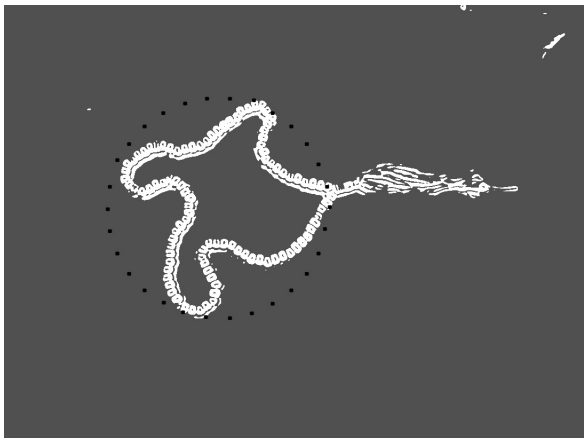
Problem 6

At this problem, I have implemented $ActiveContour$ for the following image.

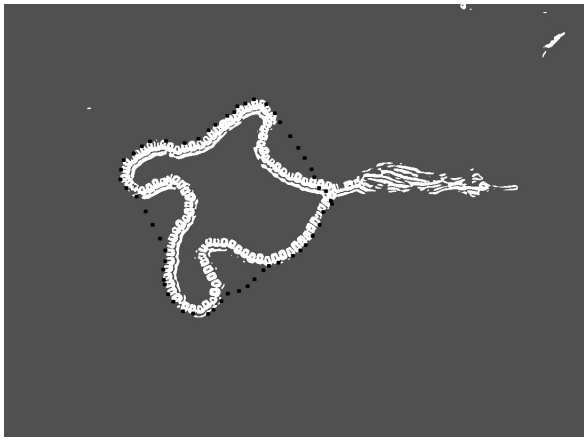
For doing this, first I found its edges by running the *edge.py* code. The result is as follows.



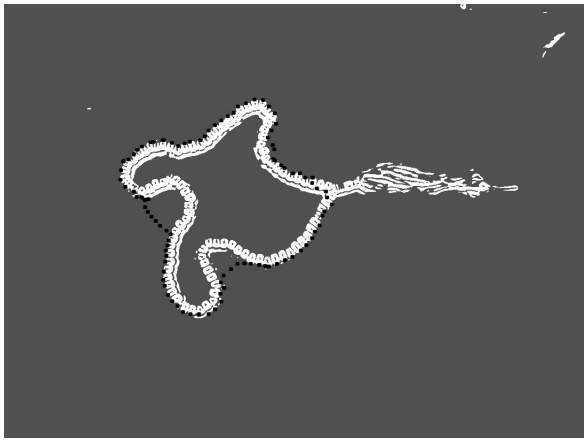
Then I have considered a circle around the given image, which has 30 points and is as follow.



Then after some iterations and moving points, when they have stoped moving, I doubled the points.



And after 150 iterations the result is as follow.



You can find the generated video as *movie*01.*avi* at the directory.

Describe functions

1. *active_contour*: This function finds the active contour in 150 iterartions and the searching box near each point contains 25 pixels, after a number of iterations It would contain 9 pixels.
2. *double_snake*: It almost doubles the number of points of the snake.
3. *get_image_gradient*: It calculates pixels' gradients of the image needed for $ExternalEnergy$.
4. *get_total_energy*: It calculates total energy of the givven point.

The code is in *Q*6.*py*, for runing the code at first run *pythonedge.py* and then run *python Q*6.*py*.