CS 7690, Advanced Image Processing

Project6 Normalized Graph Cuts

Xiang Hao

The implementation of the affinity matrix and GC and NGC algorithms

Affinity matrix

Given a distance matrix, I use a Gaussian function, $\exp(-distance^{٢}/(٢\sigma^{٢}))$, to calculate the affinity matrix. In the Gaussian function, the sigma is very important. In addition, we also need a threshold when calculating the affinity matrix.

If there are more than one distance, say intensity distance and position distance, I will compute the affinity matrix for each distance and then multiply them together as the final affinity matrix.

GC algorithm and NGC algorithm

For Graph Cut and Normalized Graph Cut algorithms, in chapter 16.4, it use a iterative thresholding method to calculate the clusters. I did not use this method, because:

- The thresholds are important.
- It is not easy to find the right thresholds when the image is not simple.
- There are other methods to calculate the clusters.

I use k-means to calculate the clusters.

GC algorithm:

Given the affinity matrix, do the Eigen Decomposition.
Choose the first K eigenvectors corresponding to the largest eigenvalues.
Treat the K eigenvectors as the feature matrix, do a k-means clustering on the feature matrix.

NGC algorithm:

Given the affinity matrix, compute the normalized matrix as $D^{-1/2}(D-A)D^{-1/2}$
Do the Eigen Decomposition of the normalized matrix.
Choose the first K eigenvectors corresponding to the smallest eigenvalues except the first smallest one.
Treat the K eigenvectors as the feature matrix, do a k-means clustering on the feature matrix.

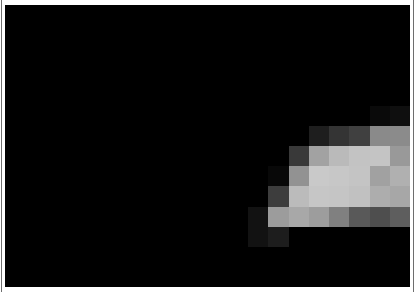
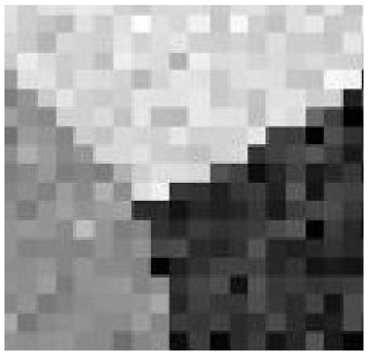
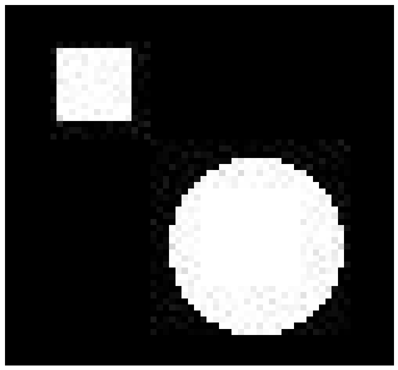
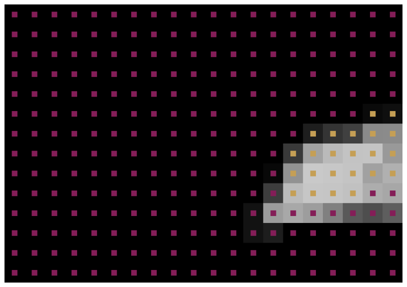
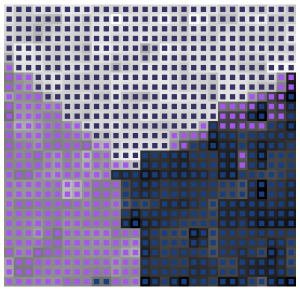
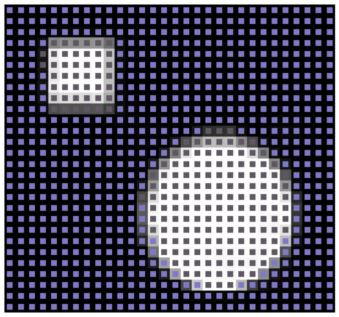
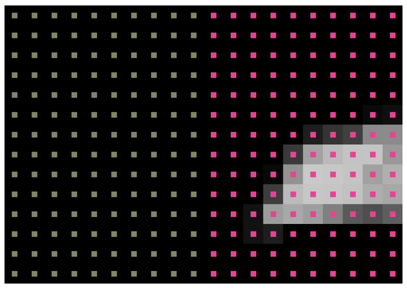
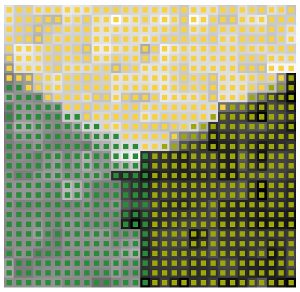
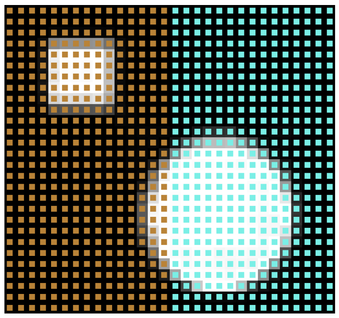
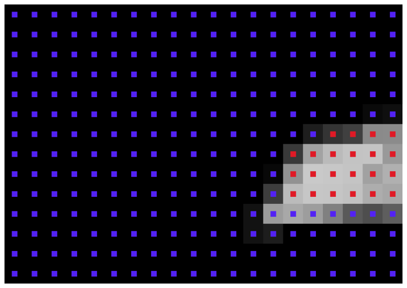
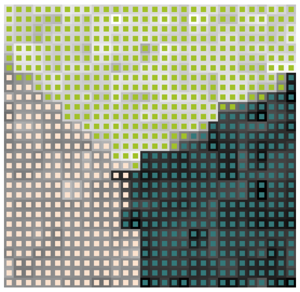
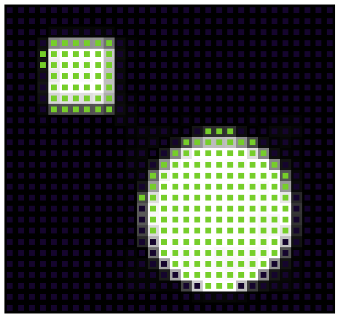
K-means algorithm:

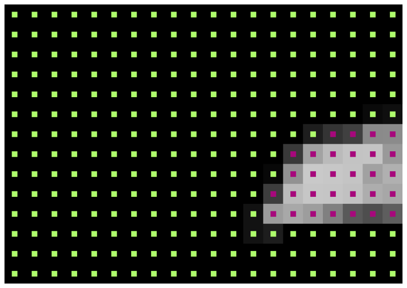
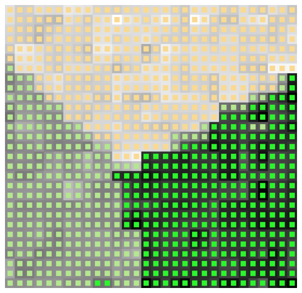
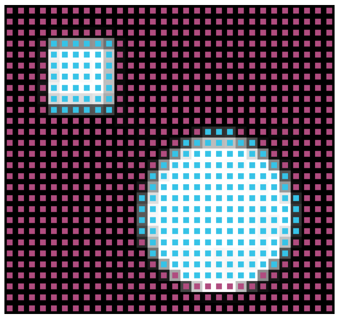
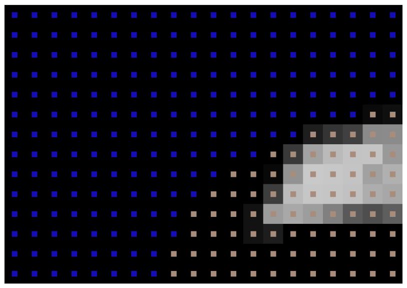
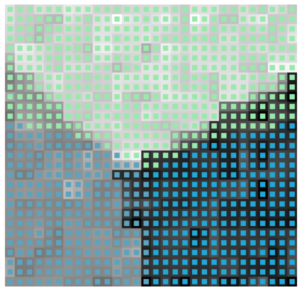
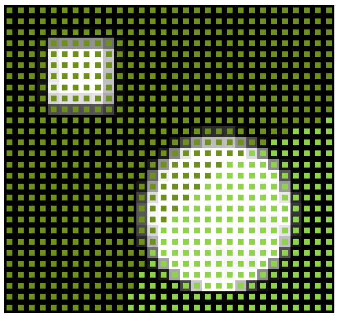
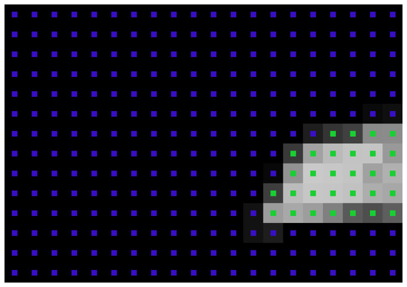
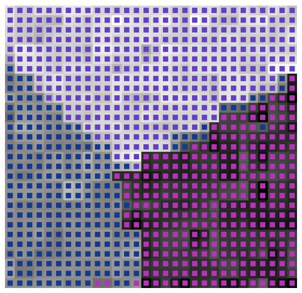
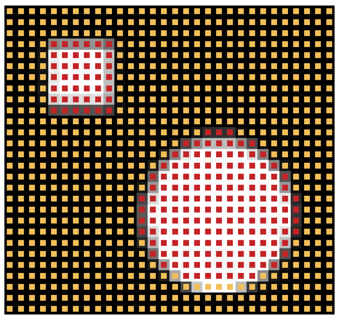
- Initialize the center of the clusters
  I implement three different methods:1) choose the first k points as the centers 2) randomly choose k points as centers 3) ldh: first randomly choose a center, and then choose the points with largest distance from the exist centers.
  I usually the ldh method to initialize the centers.
- Assign each data point to the closest center
- Recompute the centers as the means of the data points
- If assignments have changed, go to 2

Description of the success of segmentations

Demonstration:

In the following demonstration, I adjust the sigma parameters and show the optimal results I got.

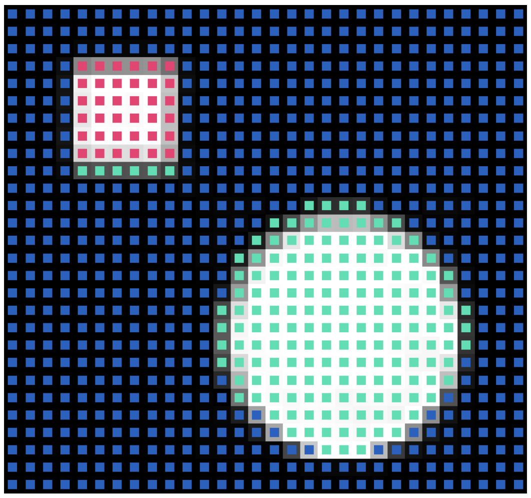
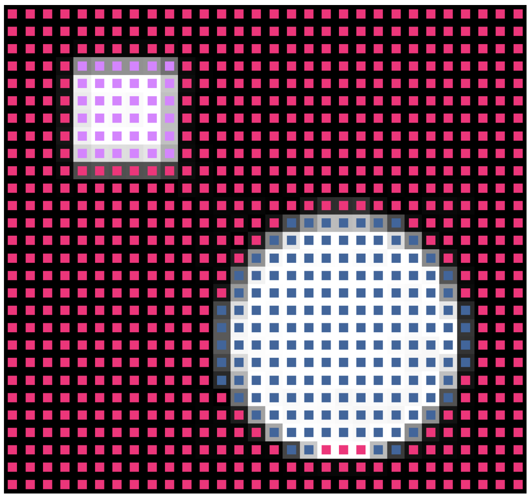
	Image1	Image2	Image3
Original Image			
GC-Intensity			
GC-Distance			
GC-both			

NGC-Intensity			
NGC-Distance			
NGC-both			

From the above tests, we can see:

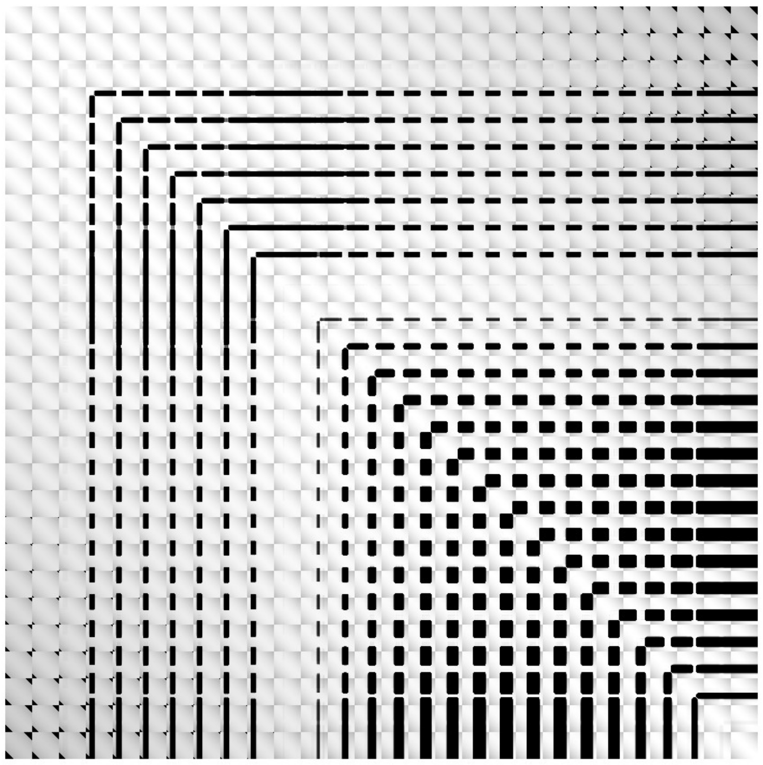
1)Normalized Graph Cut may be better than Graph Cut, but sometimes, it is not better than Graph Cut. It's a problem dependent.

2)For simple images, for example, there are only two clusters, distance may not be useful. If the image is very complicated, for example, we want to separate two regions with similar intensities, distance will help. See the following example.

GC	NGC
	

In the above example, I am trying to separate the rectangular and the circle using intensity and distance affinity measurements. These two regions have very similar intensities. Without using the distance affinity measurements, I cannot get the above results.

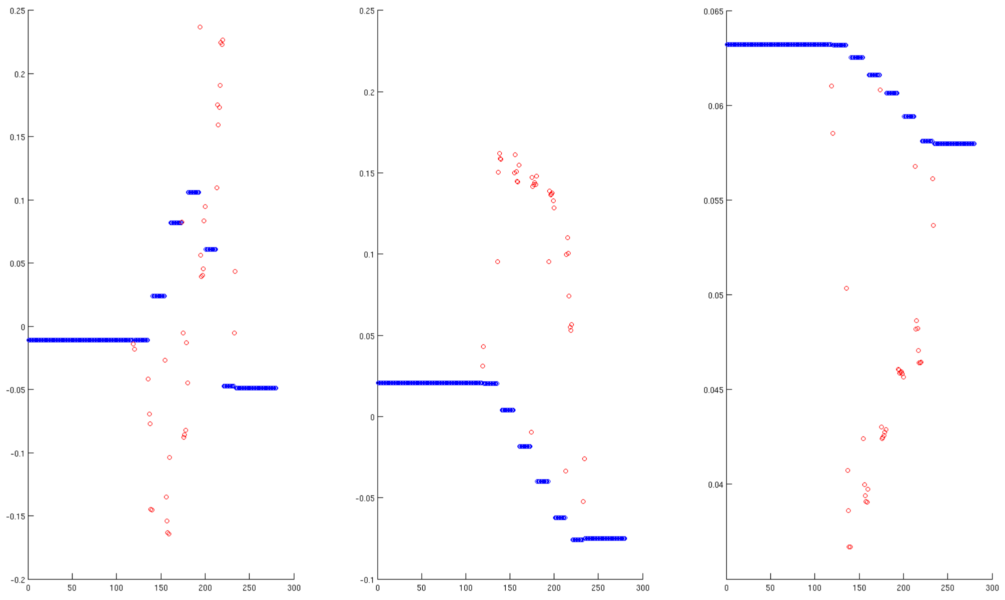
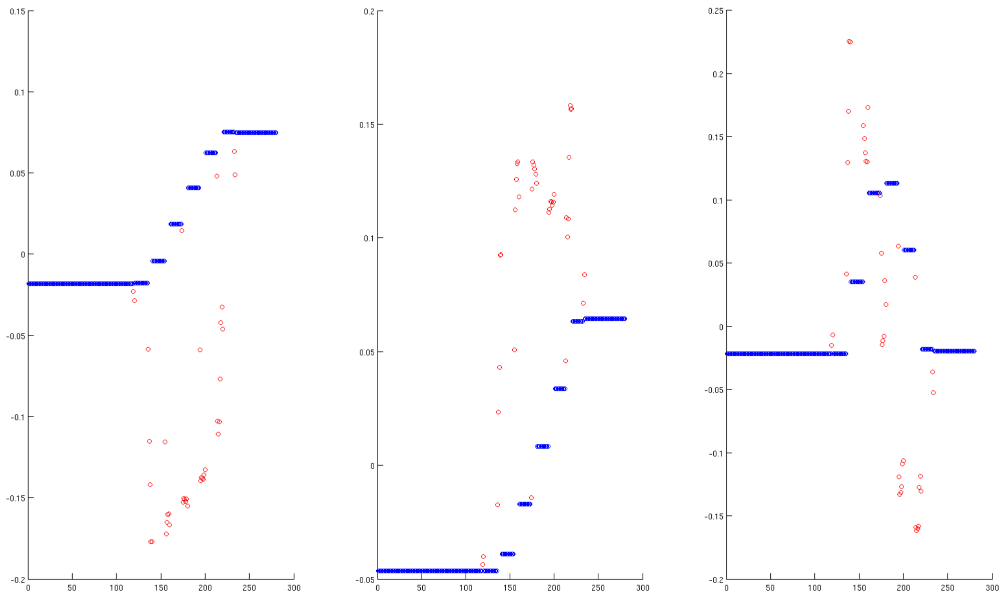
Display the affinity matrix



This is a affinity matrix for image3. I use this matrix to separate the rectangular and the circle from the background. In the above image, we can clearly see there are two regions with black stripes. These two regions are the rectangular and circle.

Display the eigenvectors

GC vs. NGC

GC-Intensity	
NGC-Intensity	

The above images is the eigenvectors, which are used to segment the image1 with intensity affinity measurements. There are two clusters in the image. The red points are in one cluster and the blue points are in the other cluster.

Compare the eigenvectors of GC and NGC,

- If we only look at the first eigenvectors, NGC is better than GC since the red points are more separable in the eigenvector of NGC.
- If we only look at the second eigenvectors, there are little difference. GC did a little better job than NGC.
- If we only look at the third eigenvectors, GC is much better than NGC.

(When I say the first eigenvector, I actually mean the eigenvector corresponding to the largest eigenvalue in GC or the eigenvector corresponding the second smallest eigenvalue in NGC.)

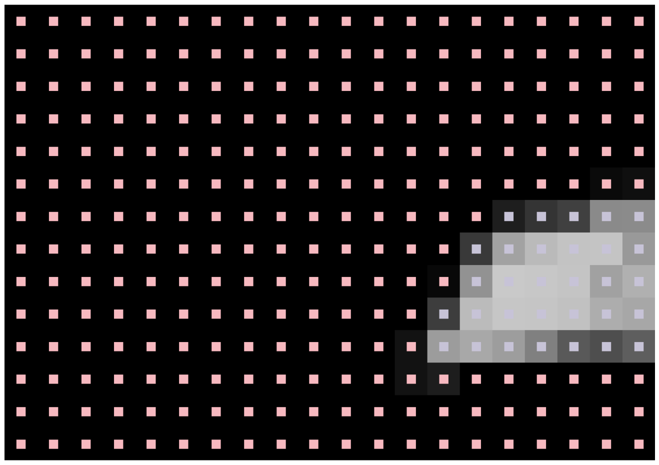
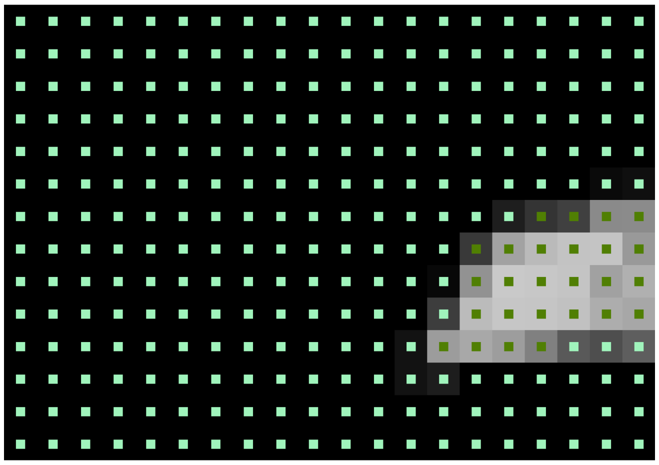
This leads to several questions:

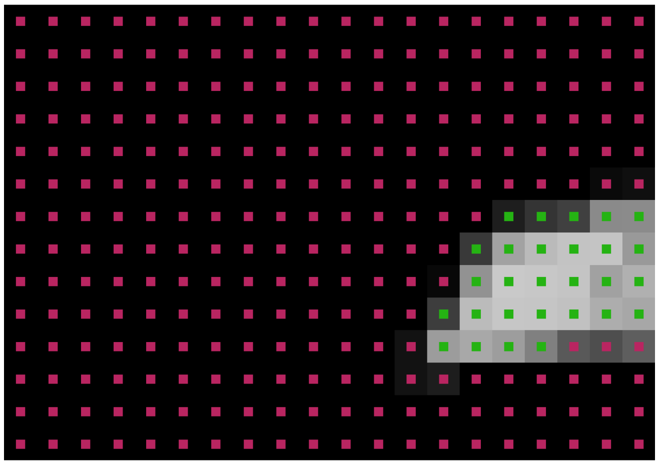
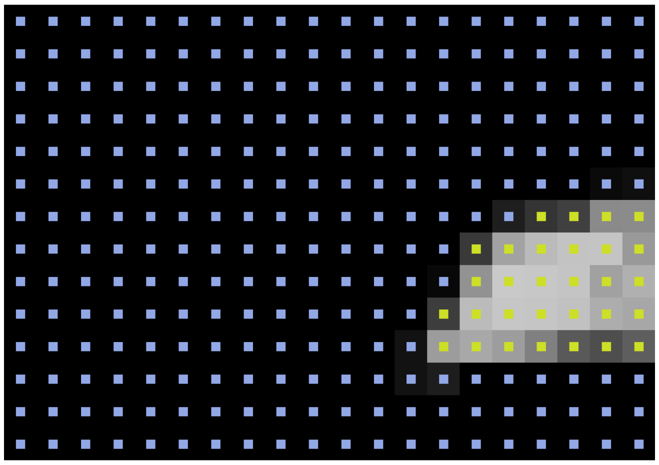
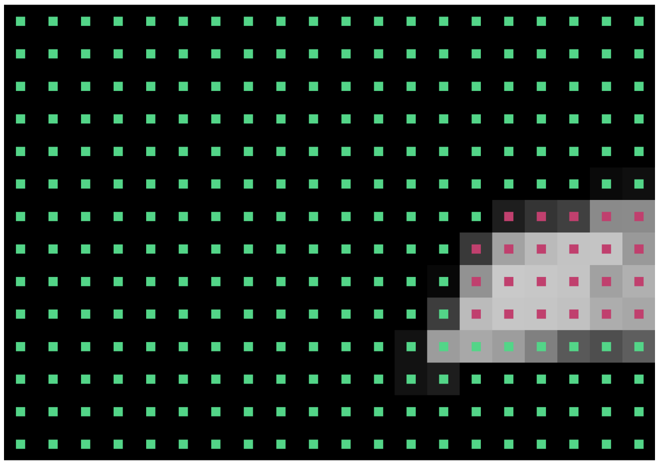
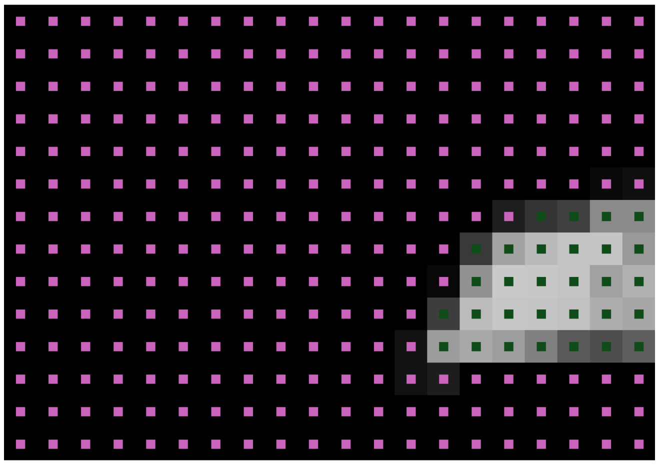
1) How many eigenvectors we should use?
2) Which eigenvectors we should use?

In order to understand the above two questions better, I do a test here:

From the above discussion, we have known:
In GC, eigenvector3 > eigenvector2 > eigenvector1
In NGC, eigenvector1 > eigenvector2 > eigenvector3

I will respectively use 1)the best eigenvector 2) the best and the second best eigenvectors 3) all three eigenvectors to do the clustering and compare the results.

GC	NGC
	
Only use the third eigenvector	Only use the first eigenvector

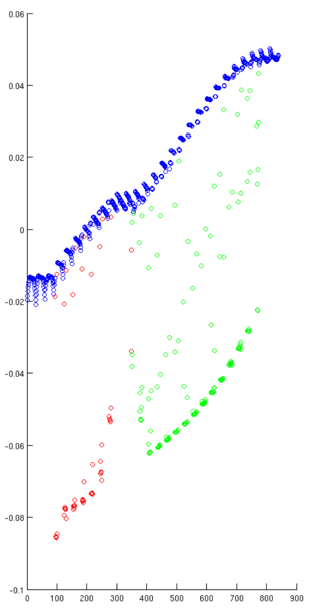
 Use the third and second eigenvectors	 Use the first and second eigenvectors
 Use all three eigenvectors	 Use all three eigenvectors

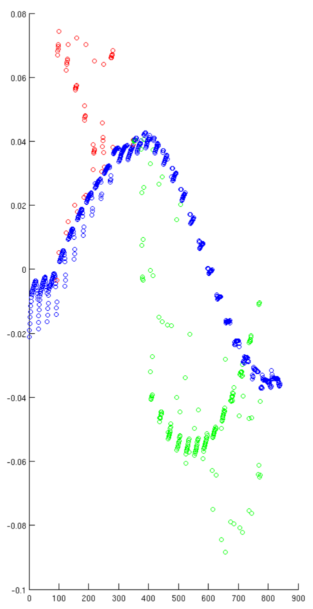
From the above comparison, we can see,

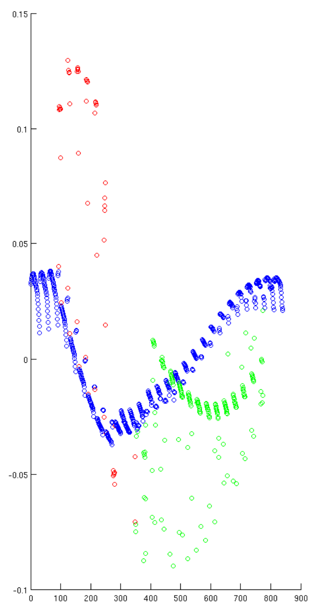
- GC get the best results when it only use the third eigenvector. NGC gets the best results when it use first and second eigenvectors or use all three eigenvectors.
- In GC, when we use the third and second eigenvectors, the result is worse than the one only using the third eigenvector.
- GC's best results is better than NGC's best results.

The above test prove that the GC or NGC have problems of eigenvectors selection, but it did not solve the two questions. In order to solve it, we may need to use some knowledge of feature selection technology.

A linter Combination of eigenvectors:







The above image is the eigenvectors, which are used to separate the rectangular and the circle from the background in image3. The red points are in the rectangular; The green points are in the circle; The blue points belong to the background.

From figure1, the rectangular and the circle cannot be fully separated from the background, but figure2 and 3 can help to separate some of the points.
This is very similar with the problem discussed in figure 16.23: Though a single eigenvector cannot suggest a good clustering, but a linear combination of the eigenvectors can lead to a good clustering.

The same thing happened when we try to separate the rectangular and the circle, which is equivalent to separate the red points and the green points in the above figures. By combining the eigenvectors, we can get a good clustering.

Critical discussion

Problem1:
The Graph Cut and Normalized Graph Cut are good algorithms, but a major problem is that the algorithms is very sensitive to the parameters.

Even I changed the parameters a little, the clustering results may change a lot.

Problem2:
Another problem is that how many eigenvectors we should use. On one hand, if we choose very few, we may not have enough information to get a good clustering. On the other hand, if we use too many, the clustering algorithms may feel confused.

Problem3:
Time cost. This algorithm costs a lot of time even for a small size image. Suppose we have a large image database, it will be very hard for use to use Graph Cuts to do image segmentation.

In this implementation, I use k-means to calculate the clusters, but we can also use other methods, such as GMM.