Qidi Han (6814891757)
EE-569
Homework #4
Mar-19-2020

Problem 1: Texture Analysis and Segmentation

I. Abstract and Motivation

In many models of image processing, the texture classifier and segmentation are very important part. There are many different texture analyses including feature extraction. Before, we do analysis, the first main part is feature extraction, and then we have texture classification. There are many technical that will used in this problem. The feature extraction will use laws filter. Texture classifier will help us to determine the objects. Also, this technical can determine how many different texture are included in one image.

II. Approach and Procedures

a. Texture Classification- Feature Extraction

The first half of the texture analysis is feature extraction. The process in use the 5*5 laws filters. The laws filter table 1 shows below.

Table1

Name	Kernel
L5(Level)	[1 4 6 4 1]
E5(Edge)	[-1 -2 0 2 1]
S5(Spot)	[-1 0 2 0 -1]
W5(Wave)	[-1 2 0 -2 1]
R5(Ripple)	[1 -4 6 -4 1]

The first step is building the 25 laws filters. Each filter is built by two vectors proved above. For example, the first laws filter will be L5'E5 = [-1 -2 0 2 1; -4 -8 0 8 4; -6 12 0 12 6; -4 -8 0 8 4; -1 -2 0 2 1];

Before the process, the pro-processes is the image will subtract the image mean to reduce the illumination effects. Then do the boundary extension for the image, since our filter is 5*5. Our boundary extension will 4 more rows and 4 more columns. Then apply 25 laws filters with our boundary extension image. We will have 25 filtered images. Then I will use the formula below to calculate the energy with the whole image, x and y represent the size of the matrix.

$$each_{feature_{energy}} = sum(abs(pixel))/(x*y) \quad (1)$$

After calculating each feature energy, we will 25-dimension feature vector. Then I will use table below to reduce the 25-dimension feature vector to 15-dimension feature vector, because there are some feature energy that can be combined together.

Tabel2

L5L5	(L5E5+E5L5)/2	(E5S5+S5E5)/2	E5E5	(L5S5+S5L5)/2

(E5W5+W5E5)/2	(S5S5)/2	(L5W5+W5L5)/2	(E5R5+R5E5)/2	W5W5
(L5R5+R5L5)/2	(S5W5+W5S5)/2	R5R5	(W5R5+R5W5)/2	(S5R5+R5S5)/2

Then, we will repeat the same process with 36 training samples. We will have 36*15 matrix. 36 represents 36 images, 36 means each image have 15 different feature vectors. After I do the 36 training images, I will do the same process with test images.
Then I will calculate the feature dimensions has the strongest discriminant power and weakest discriminant power. The process of discriminant power is I calculate the inter class average for each feature first. The formula proved below. $each\_class\_average$ represent the mean of each class, there are four class for our question. $total\_image\_average$ represent the mean of those 36 images.

$$inter\_eachfeature = sum(each\_class\_average - total\_image\_average)/(4) \qquad (2)$$

The equation for the intra class sum is proved below. $eachfeature$ represent the each feature data point, $ave\_feature$ represent the average for the class. There are 9 images for each class. Therefore, we have to sum each feature for each class and divide by 9. And There are four image class for our question, therefore, the inter class for each feature is sum of all image feature and divided by 4. Figure 1 shows the result for training data.

$$inter\_eachfeature = sum(sum(eachfeature - ave\_feature)/9)/(4) \qquad (3)$$

Before the PCA, I also normalize the feature data. Since L5L5 had non-zero mean, I will divide L5L5 to all the features.
(PCA Bounds)
Finally, I will reduce the 15 features space to 3 features space. For the PCA process, I write the function in the MATLAB, the process of the PCA is calculating the mean for each features firstly. Then subtract the feature mean to get the zero-mean matrix. Then use the svd() function in the MATLAB to get the singular values in descending order. Since we want to reduce to 3 dimensions. We keep the top 3 components. And use this top 3 components to calculate the reduced matrix Yr=X*Vr. X is the original data matrix. Vr represent the top 3 components of the singular values.
After I reduce the matrix to 3-D feature vector for the training and test data, I will plot the feature vector in the feature space. Figure 2a shows the training data in feature space.

b. Advanced Texture Classification --- Classifier Explore

After, I extract the features from the image, I will classifier those features. The first method is called Unsupervised, and the second method is called supervised.

- Unsupervised

For the Unsupervised case, we will use k-means for classifier different categorize of textures. The K-means are only need test data. For both 15 dimension and 3 dimensions, I will use k-means to get the answer. I will compare the result with observations. Figure 3 shows the result for the classifier with observations. Finally, I will calculate the error rate for both 15 dimension and 3 dimensions. Table 3 shows the error rate for both 15 D and 3 D.
(K-mean Bounds)
I make the function for the k-mean in my project. The first step of the k-means is initializing the mean data from k samples. For our question, we will have 4 class. I will randomly choose 4 data

to become our first means. Then I will use the norm to calculate the distance between each point, I will set the data to k class if the norm of that this data point to the mean is closest. After, I process all the data point, I will re-calculate the means for each class. Finally, repeat the same process above, the iterations are user input, I usually input 1000 iterations. There are some issues of my k-means, I will discuss it in my discussion part. Also, I will discussion the different with MATLAB kmeans function with my K-mean function. Figure 3b shows the unsupervised result by 15-D features and 3-D features.

- Supervised

For the supervised learning, there are two different type of the learning. The first called Random forest and the second one called Support Vector Machine. The both need train the data first, then use the training data point to classifier all test data. I will compare two kinds of classification in the discussion part. Table 4 shows the error rate for both 3D feature vector for RF and SVM. As random forest, I had mentioned in the project 2. We will have many decision trees in our random forest. I will run 80 decision trees in my random forest. After I build the random forest, I will use RF.predict() function to predict the result for our test data. For the SVM, we will have two different ways to do the question, the first way is called one vs one. We use one type data points to compare with other one type data points. And draw the decision boundary by using only two type of data points. Then we will use 4 times in our question, since we have 4 class. Finally, I will input the test data into our decision boundary, the data will belong to one class with the nearest decision boundary. But there is an issue of those method, it's one vs one will give us an undetermined region. This is not satisfying our problem; therefore, I will use one vs rest to the question. The one vs rest technical is that I only use one data point to compare with the rest data point to draw the decision boundary, there are no undetermined region for this technical. Therefore, it satisfies for our question, and I will use fitcecoc() function to do our question. Figure 3c shows the supervised result by RF. Figure 3d shows the supervised result by SVM.

c. Texture Segmentation

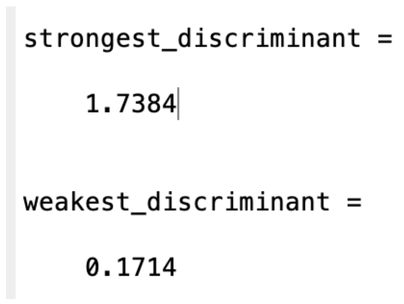
There are four big steps for our texture segmentation, the first step is feature extraction, I will do the same step with last question, I apply the laws filters to the input image, and reduce 25 D to 15 D. Before, I reduce the 15 D, I do the energy feature computation, I set a window and use equation 1, but the x and y are the window size for this question. Then, I normalize the feature energy by divide L5'L5. Next, I reshape the 450*600 matrix to 270000*1 for each image. Finally, I will combine 15 images together to get 270000*15 matrix. Then I use k-means function to classifier each pixel, and set each class to (0, 51, 102, 153, 204, 255). And reshape the image back to 450*600. Figure 4 shows the final image of texture segmentation with windows size equal 13 15 21 31 35 51.

d. Advanced Texture Segmentation

Since our result in Figure 4 is not perfect, I designed a way to perform a better solution. Before the PCA step, I will use the same process with last question. Figure 4dd shows the image with window size equal 35 and PCA with dimension 8. The next step takes out each part from the original image. And save the part with largest area. I use the same method with last assignment. I

will label all the continues area with same label and save the large area with the greatest amount of pixels. Figure 4d_1 shows the first part of the reminding image. Figure 4d_2 shows the second part of the reminding image. Figure 4d_3 shows the third part of the reminding image. Figure 4d_4 shows the fourth part of the reminding image. Figure 4d_5 shows the fifth part of the reminding image. Figure 4d_6 shows the sixth part of the reminding image.
Then I will combine all the part into one big image, and set each class to (2, 51, 102, 153, 204, 255). Figure 4d_7 shows the combination image.
The final step is making the dark area with their neighbor pixel. Figure 5 shows the final image after I make the dark area with the surrounding pixel values.

III. Experimental Results



```
strongest_discriminant =

    1.7384


weakest_discriminant =

    0.1714
```

Figure 1 shows the result for training data.

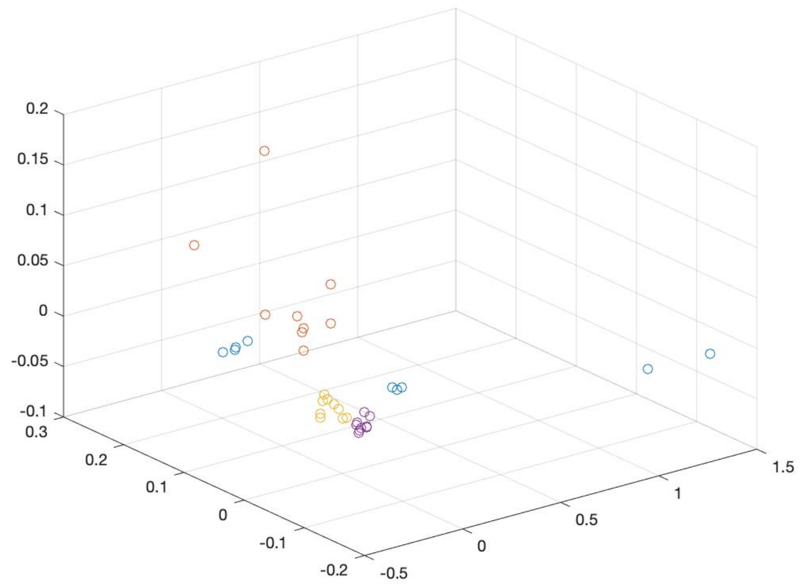
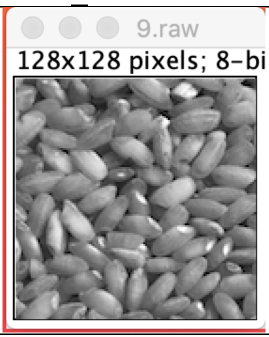
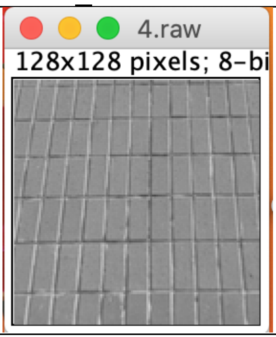
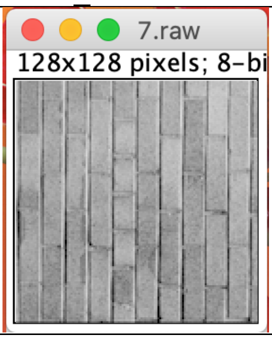


Figure 2 shows the training data in feature space (3D space).

Class: Grass	Class: Grass	Class: Grass
File_name: 1.raw	File_name: 6.raw	File_name: 12.raw

Class: Rice	Class: Rice	Class: Rice
File_name: 5.raw	File_name: 8.raw	File_name: 9.raw
		

Class: Brick	Class: Brick	Class: Brick
File_name: 4.raw	File_name: 7.raw	File_name: 10.raw
		

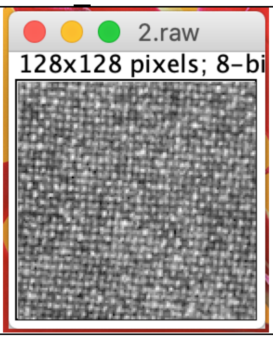
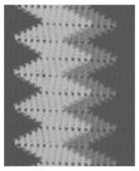
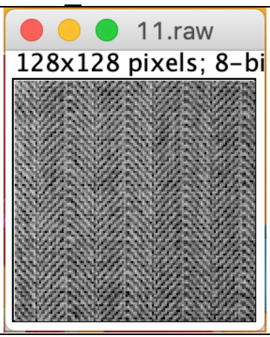
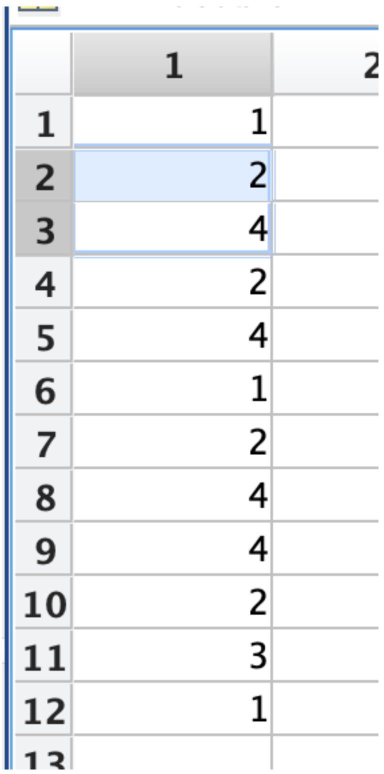
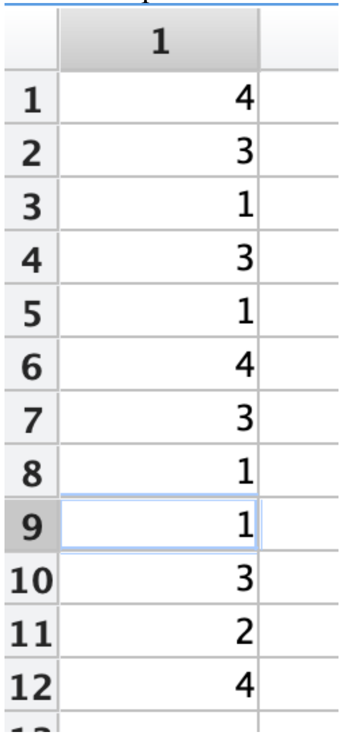
Class: Blanket	Class: Blanket	Class: Blanket
File_name: 2.raw	File_name: 3.raw	File_name: 11.raw
		

Figure 3 shows the result for the classifier with observations.



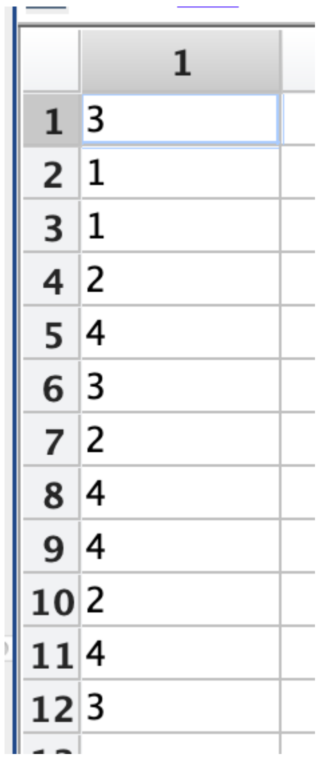
	1
1	1
2	2
3	4
4	2
5	4
6	1
7	2
8	4
9	4
10	2
11	3
12	1

Figure 3b_1 shows the unsupervised result by 15-D features.



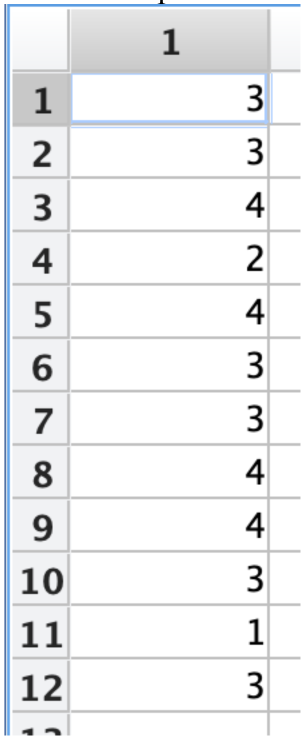
	1
1	4
2	3
3	1
4	3
5	1
6	4
7	3
8	1
9	1
10	3
11	2
12	4

Figure 3b_2 shows the unsupervised result by 3-D features.



	1
1	3
2	1
3	1
4	2
5	4
6	3
7	2
8	4
9	4
10	2
11	4
12	3

Figure 3c shows the supervised result by RF.



	1
1	3
2	3
3	4
4	2
5	4
6	3
7	3
8	4
9	4
10	3
11	1
12	3

Figure 3d shows the supervised result by SV

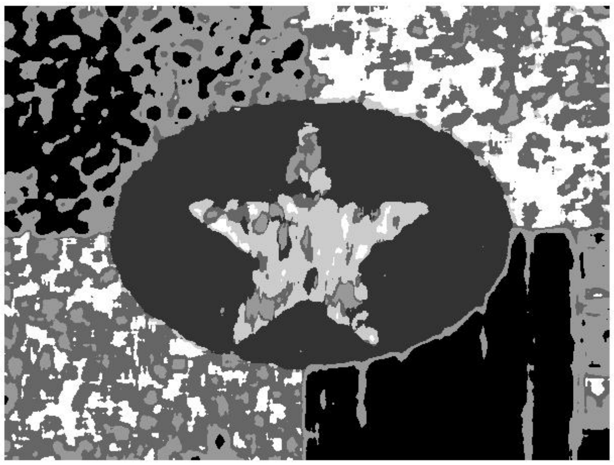


Figure 4a shows the final image of texture segmentation with window size equal 13

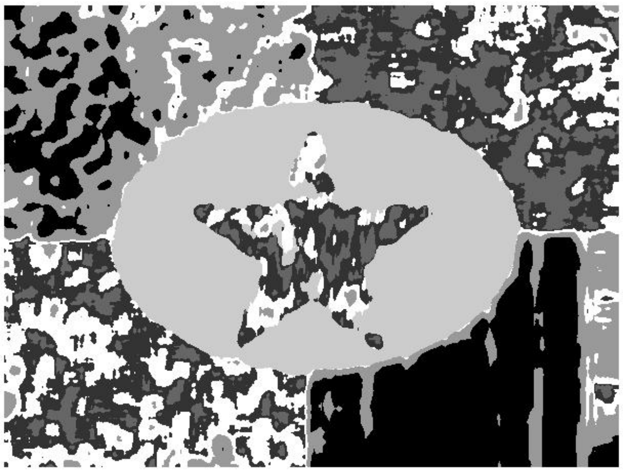


Figure 4b shows the final image of texture segmentation with window size equal 15

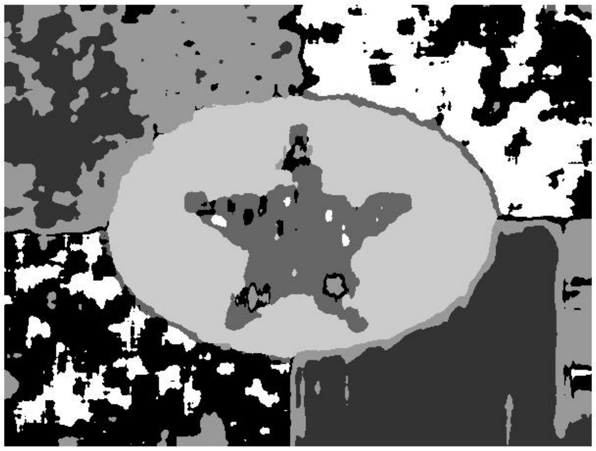


Figure 4c shows the final image of texture segmentation with window size equal 21

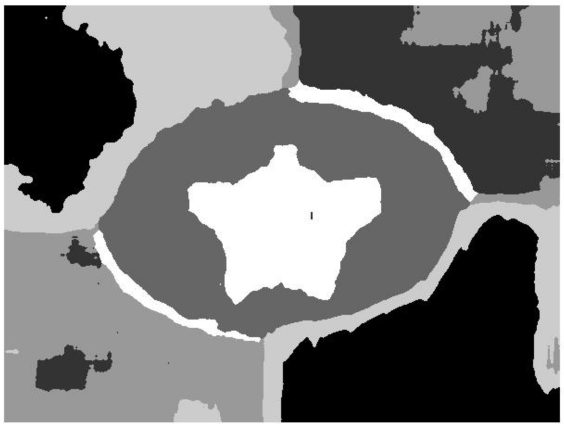


Figure 4d shows the final image of texture segmentation with window size equal 31

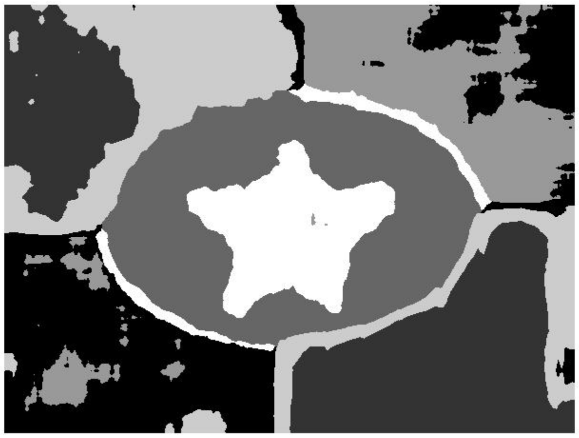


Figure 4e shows the final image of texture segmentation with window size equal 35

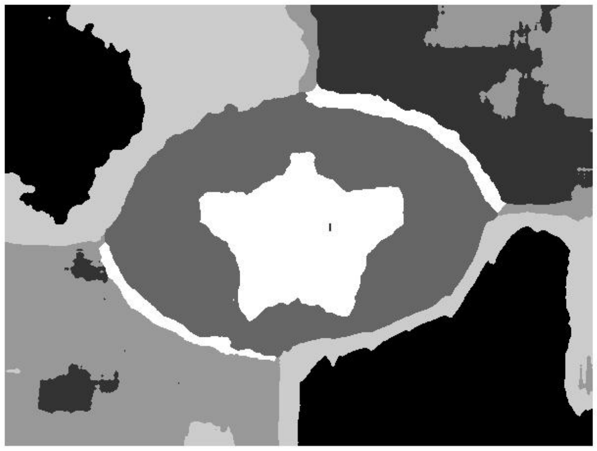


Figure 4f shows the final image of texture segmentation with window size equal 51

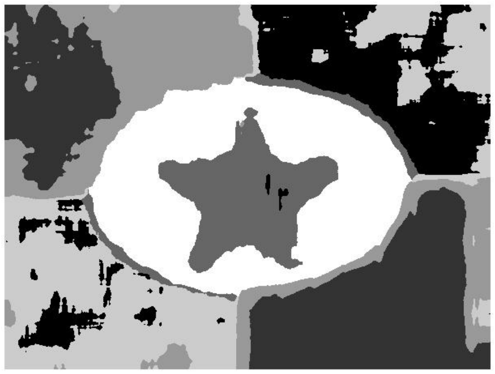


Figure 4dd shows the image with window size equal 35 and PCA with dimension 8

Figure 4d_1 shows the first part of the reminding image.

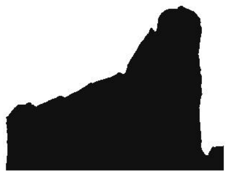


Figure 4d_2 shows the second part of the reminding image.

Figure 4d_3 shows the third part of the reminding image.

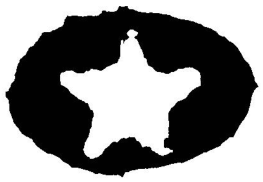


Figure 4d_4 shows the fourth part of the reminding image.

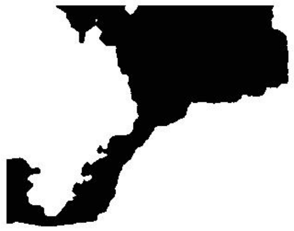


Figure 4d_5 shows the fifth part of the reminding image.

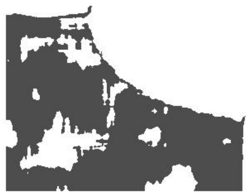


Figure 4d_6 shows the sixth part of the reminding image.

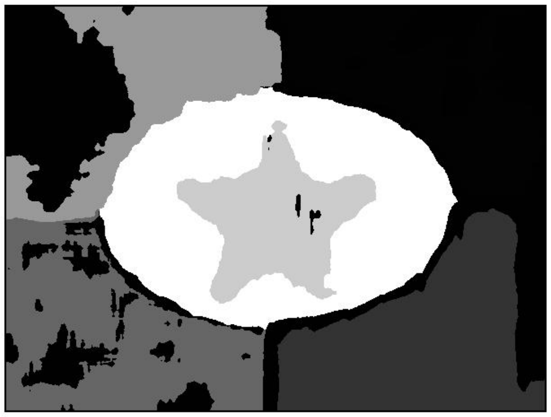


Figure 4d_7 shows the combination image.

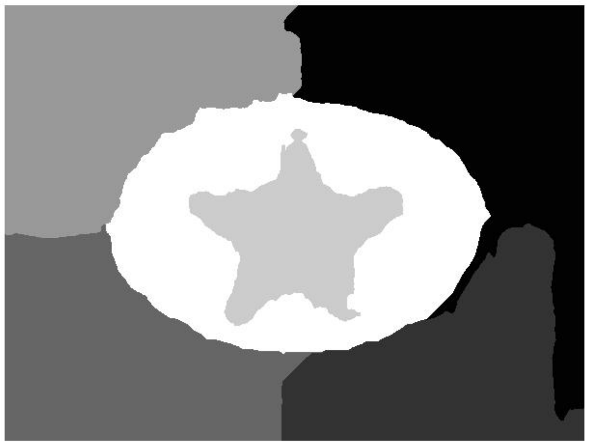


Figure 5 shows the final image after I make the dark area with the surrounding pixel values.

IV. Discussion

1.a-1
I used boundary extensions with additional 4 rows and 4 columns, since our filter size is 5*5
1.a-2
For the training data: The strong discriminant power is belonging to feature E5E5(Figure 1). The weakest power is belonging to feature L5S5/S5L5(Figure 1).
1.a-3
Figure 2 shows the result.
1.b-1
For the 15-D: Compare Figure 3b_1 with Figure 3, we have two classes are misclassified. Two images from the blanket are wrong. For Figure 3b_1, ‘1’ represent grass. ‘4’ represent rice. ‘2’ represent brick. ‘3’ represent blanket. The error rate is 2/12=0.1667

For the 3-D: Compare Figure 3b_2 with Figure 3, we have two classes are misclassified. Two images from the blanket are wrong. For Figure 3b_2, ‘4’ represent grass. ‘1’ represent rice. ‘3’ represent brick. ‘2’ represent blanket. The error rate is 2/12=0.1667

The effectiveness:
From my perspective, I think there is no change for the dimension reduction, because there are both 2 class are misclassified, but from classification views, 3D should better than 15D, because there is many useless information or there are many useless features in 15D. therefore, when we do the feature reduction, 3D should give us a better result. 3D gives us the most important information or the most important features. Every time, I run the k-means, it will give me different answers, I think it depend on the initial means, or the initial vectors. I use the random() function to choose the initial vectors, therefore, the number will be different for each case.

1.b-2
Figure 3c and Figure 3d show the result for the random forest and SVM result. For the random forest method, we have only 1 missed class. Image 11 is misclassified, ‘4’ represent rice. ‘1’ represent blanket. ‘3’ represent grass. ‘2’ represent brick. The error rate is 1/12=0.08333
For the SVM, there are 4 missed class. ‘4’ represent rice. ‘1’ represent blanket. ‘3’ represent grass. ‘2’ represent brick. The error rate is 4/12=0.33333
Comparison:
From our result, random forest method is better, but two method are totally different, random forest is used to extract features and use those features to build each decision tree and combine all the decision tree to a random forest. The number of decisions tree also affect the

results. But SVM use the ‘distance measure’ (margin) to classifier the class. I think for different data set, the answer for SVM and random forest will be different, I cannot talk with method is better, I only can see that random forest is better in our case, because extracting features is the most important thing in our question, therefore random forest is slight better.

1.c-1
Figure 4a(windows size = 13), Figure 4b(windows size = 15), Figure 4c(windows size = 21), Figure 4d(windows size = 31), Figure 4e(windows size = 35), Figure 4f(windows size = 51) shows different windows size images.
1.d-1
I ready fully describe the method in the approach section step by step. Figure 4dd to Figure 5 shows the processed image.

Problem 2: Image Feature Extractors

I. Abstract and Motivation

From the last question, we already do the feature extraction, but we did in the high dimensional form. There are many technical that using to extract the feature. Use those features, we can observe some characters of the objects. When people want to compare the different image with same object, we can use this feature extraction to do the process. For this project, we will use the four different images to do the comparison with different husky. There is a library that called vlfeat. There is some important function will be used in our project.

II. Approach and Procedures

There are three parts of this experiment. The first question is answering the five question after I read the article. I will answer the question in the discussion part.

1. Salient Point Descriptor

The answer is in the discussion part.

2. Image Matching

First, I will use imread() function to read the jpg image. Then I will use vl-sift() and rgb2gray function to extract the features. I will select the largest scale of image 3. The index of largest scale is 1089. I will use this point to the matching part. Next, I will use vl_ubcmatch() function to match the largest scale features in image 3 with whole image 1. I will use vl_plotframe () function to plot the largest scale point with must match point in figure2. Figure 6a shows matching point from figure 3 and figure 1 when I use sum function. Figure 6b shows matching point from figure 3 and figure 1 when I use norm function. The formula for sum is showing below.

$$distance = sum((image1\_descriptor - image2\_descriptor).^{\wedge}2) \qquad (4)$$

The formula for norm is showing below.

$$distance = norm((image1\_descriptor - image2\_descriptor)) \quad (5)$$

The second question of this part is showing the SIFT pairs between each image. There are four images in this question. Figure 7 shows the SIFT pairs between Husky 1 and Husky 3. Figure 8 shows the SIFT pairs between Husky 3 and Husky 2. Figure 9 shows the SIFT pairs between Husky 3 and puppy 1. Figure 10 shows the SIFT pairs between Husky 1 and Puppy 1. There are many fails in our question, I will discussion the reason in the discussion part.

3. Bag of Words

The last part of this project is combining the k-means clustering with our SIFT features. There are 8 different type of feature class in our problems. I will use the same vl_sift() function to extract the features, then I will use kmeans to classifier each feature to 8 different class. Finally, I will plot those 8 different features into a histogram plot. Figure 11 shows the histogram of Husky1. Figure 12 shows the histogram of Husky2. Figure 13 shows the histogram of Husky3. Figure 14 shows the histogram of puppy1. This process is called bag of words. Figure 15 shows the comparison codewords between all four images and Husky 3 codewords. I will discuss the result in the discussion part.

Experimental Results

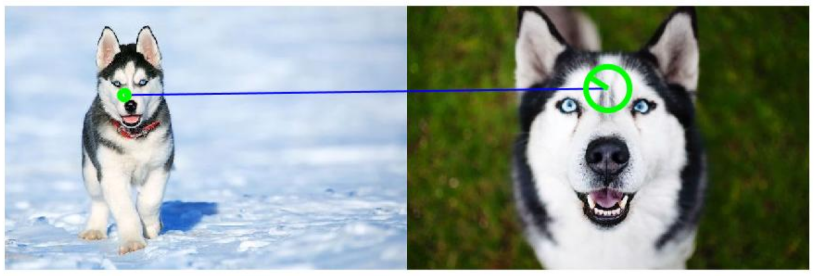


Figure 6 shows matching point with largest scale from figure 3 and figure 1. (Use the sum function)

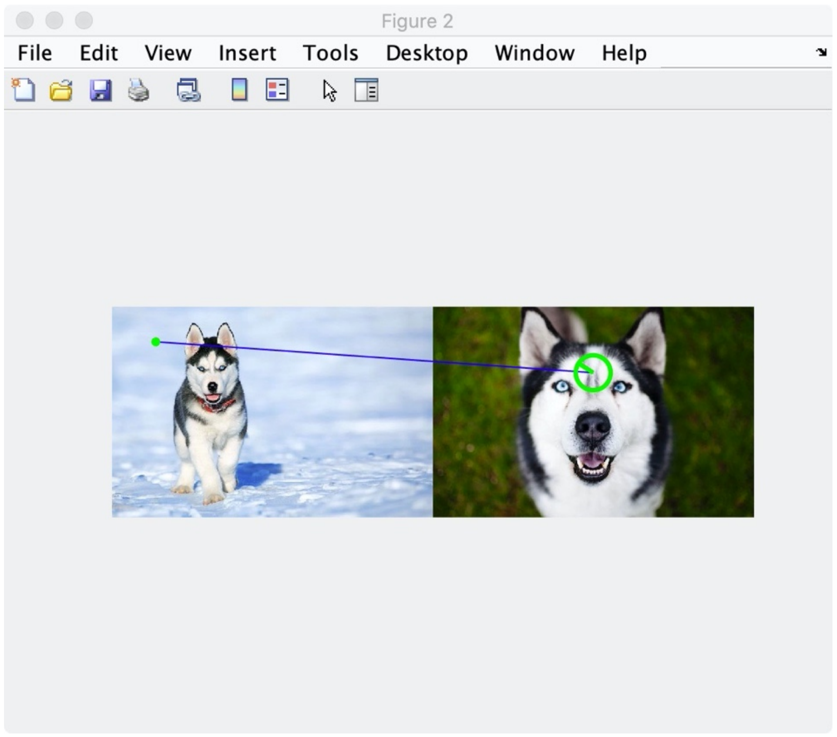


Figure 6 shows matching point with largest scale from figure 3 and figure 1. (Use the norm function)

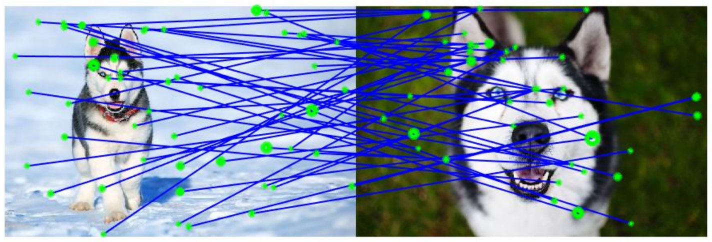


Figure 7 shows the SIFT pairs between Husky 1 and Husky 3.

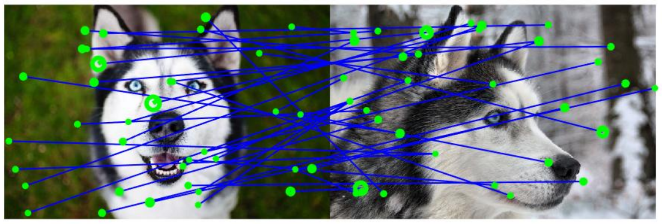


Figure 8 shows the SIFT pairs between Husky 3 and Husky 2.

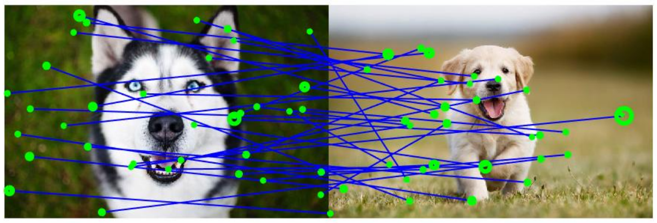


Figure 9 shows the SIFT pairs between Husky 3 and puppy 1.

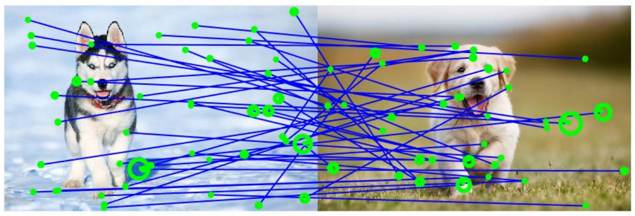


Figure 10 shows the SIFT pairs between Husky 1 and Puppy 1.

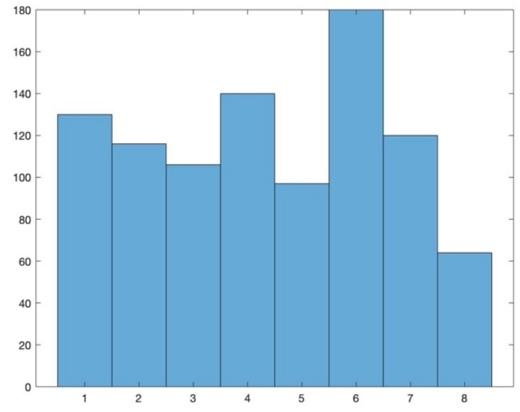


Figure 11 shows the histogram of Husky1.

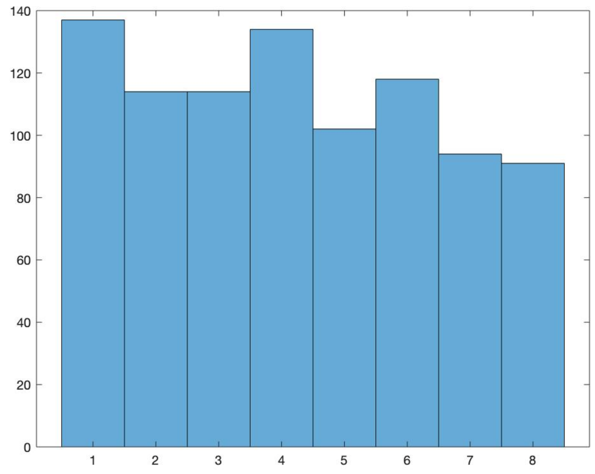


Figure 12 shows the histogram of Husky2.

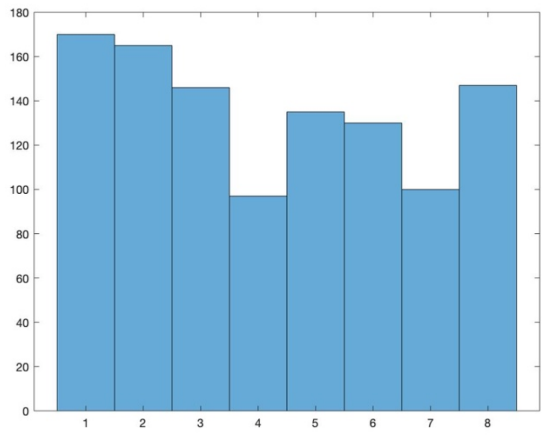


Figure 13 shows the histogram of Husky3.

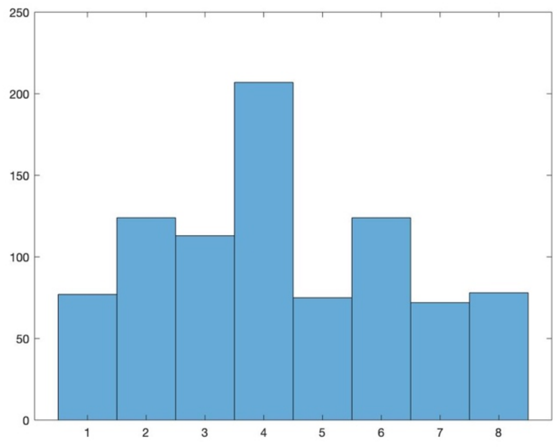


Figure 14 shows the histogram of puppy1.

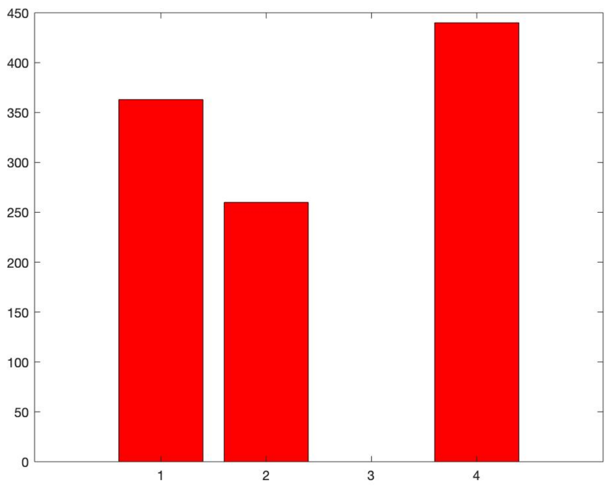


Figure 15 shows the comparison codewords between all four images and Husky 3 codewords.

III. Discussion

2a

1.

From the abstract, I learn that SIFT is robust in image scaling and rotation and finding the same object in different views of this object or background. Also, SIFT can change in the viewpoint in 3D. SIFT also can add noise to the image and change the illumination of the image.

2.

Scaling and Rotation: As the article mentioned that a least-square solution is performed for the scaling and rotation. Also, they used a fundamental matrix solution. This matrix will give us a very good solution for scaling and rotation processing. For the scaling, it also affect by the DoG, because the process of DoG require the scaling of the images. As the last assignment, we use the affine transformation, it also used in SIFT to do the scaling and rotation process.
Illumination change: I will discuss in next question.

3.

- The first reason is that each pixel value will multiple a constant number with same number that multiple in the gradients. Therefore, it will reduce the illumination change.
- The second reason is that they will set a threshold to reduce the affect by large gradient magnitudes. They usually set the threshold no larger than 0.2, and after the threshold process, they will normalize the image again. Therefore, the threshold also will reduce the illumination change

4.

Difference of Gaussians is similar with Laplacian of Gaussians. But DoG have a factor (k-1) in the equation. This term will reduce the error. Help DoG to get a better result. The different is that Laplacian of Gaussians is not scale-invariant. In other word, the DoG have the term of scale-invariant. It uses for scale normalization. As the article mentioned that DoG will give us a better result.

5.

128. 4*4*8. (page 16). Also, from the MATLAB output fa, we can see that there are 128 element feature vectors.

2b-1

Figure 6a and 6b shows the result. The answer is different for nearest neighbor search. When I use sum function, it gives a better solution. The process is described in the approach section. From both images, we can see the orientation is not vertical or horizontal. It has some angles from the Husky 3 and Husky 1. It has -30 degree of the vertical line from counterclockwise.

2b-2

Figure 7 to Figure 10 shows the result. The reason of works or fails in some case is because the background affects our result, from the lecture example, we don't have the background for searching object, therefore, each searching object feature will find in the target images. But for our case, the background is the large portion of the images. Also, the object image is different, for example, the puppy 1 and husky are different object, the feature extract will give us a different result. Since the result will be fails for some case. The last reason is that the threshold for choosing the object will be different, as I mentioned in the 2a_3, the threshold will affect the illumination, and also the threshold will affect the feature attraction.

2c-1

Figure 11 to Figure 15 shows the result, From the last histogram, we can make a conclusion that Husky 3 most matching with Husky 2. And Husky 3 least matching with puppy 1. This is reasonable and readable from the eyes. Puppy 1 is absolutely less matching with Husky 3, because the features are different from Puppy 1 and Husky3.