EE569- Introduction to Digital Image Processing

Homework #4

Sourabh J. Tirodkar
3589406164
Submission Date- 22nd March 2020

Problem 1: Texture Analysis and Segmentation

a) Texture Classification- Feature Extraction

1. ABSTRACT AND MOTIVATION

   Texture Classification is the process of identification different textures in an image which becomes an example of pattern recognition system. It is so important as its application lies in every domain like medical imaging, space research, image retrieval in Computer Vision, etc. So, there needs to be specific algorithm to extract features from the image for the training data set and apply the algorithm for the test data set and predict the texture classification using feature extraction from the images.

2. APPROACH

   There's a fix steps needs to be followed for the feature extraction from the images. The training sample is given to us which has 36 samples with labels for the images as well. The test data consists of 12 images which are without labels. Our aim is to predict the textures of test images.

   Firstly, we need to extract features. Below are the steps for the same:

   1. Feature Extraction
   2. Feature Averaging
   3. Feature Reduction

   Feature Extraction

   We need to use Law Filters for this process. We have been given 1 dimensional 5 law filters. Firstly, we need to generate 25 filters using these using tensor product.

   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]

   I have generated 25 filters using following method. The example is of L5*E5' :

$$\begin{bmatrix}1\\4\\6\\4\\1\end{bmatrix} \times \begin{bmatrix}-1 & -2 & 0 & 2 & 1\end{bmatrix} = \begin{bmatrix}-1 & -2 & 0 & 2 & 1\\-4 & -8 & 0 & 8 & 4\\-6 & 12 & 0 & 12 & 6\\-4 & -8 & 0 & 8 & 4\\-1 & -2 & 0 & 2 & 1\end{bmatrix}$$

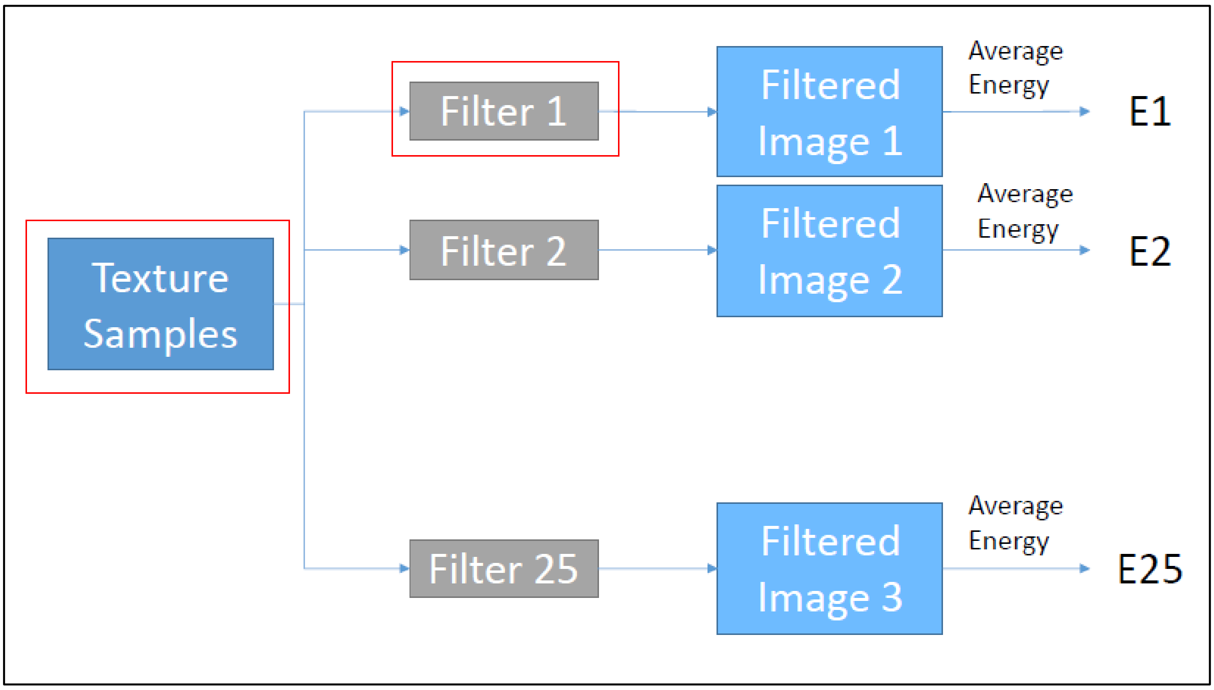
These 25 filters are taken one by one and applied on the input image which is of size 128*128 here. The filter size is 5*5 which is slide all along the image pixels. The input image is also padded with 2 rows and 2 columns at all sides. So, the input image becomes 132*132.

We get 25 filtered images for every single image. These 25 filtered images are further made into 1 single pixel by doing average energy. Each pixel is squared and sum, the total sum is divided by the N^2. Here, N=128. Thus, we obtain single value for each filter.

So far, for one input image we have 25 features (E1, E2,…..E25).

We perform all these operation for all training data set. Here we have 36 input images. We generate 25*36 matrix called feature set.

The diagram below shows the descriptive model:



<u>Feature Averaging</u>

Once we have 25D feature space, we need to reduce it to 15 dimesnional as some filter value coefficients are very close to each other. For example L5*E5' and E5*L5' does have close resemblance.

L5L5	L5E5/E5L5	E5S5/S5E5	E5E5	L5S5/S5L5	E5W5/W5E5
S5S5	L5W5/W5L5	E5R5/R5E5	W5W5	L5R5/R5L5	S5W5/W5S5
R5R5	W5R5/R5W5	S5R5/R5S5			

So the matrix size is reduced to 15*36 after feature averaging.

Feature Reduction

Our aim is to bring these 15 dimensional matrix into 3 dimensional matrix. This is done using Principal Component Analysis (PCA).
MATLAB has inbuilt pca function, but I have self implemented it using the discussion slides that were posted.
The 3D plot is plotted in MATLAB.

Feature Extraction, Averaging and Reduction are so far performed on training data set. Once done, we also perform those operation on tetsing data set to predict the labels for the test samples.

Method for PCA (Using SVD):

1. Considering feature matrix X (dim:n*m)
   n: number of samples m: number of features
2. Compute Mean mX
   mX=n∑i=1=X(i,j)/n
   i=1:n and j=1:m
3. Subtract mean to get zero mean data matrix Y_{n*m}
4. Compute SVD of Y:Y= U. S. V'
   U is n*n, S is n*m and V is n*m.
5. Sort singular values in descending order
6. Retain top k singular, where k is the no of reduced dimesnion we want. Here k=3.
7. Reduced matrix: XR= U*V(:,1:k).

Algorithm:

1. Read all training images, apply boundary extension.
2. Subtract mean of the image from every pixel to reduce illumination effects.
3. Apply 5*5 filters(total 25 filters) on each image to get 25 filtered image (for 1 image, doing it for all train:36 images)
4. Applying averaging energy to make it a singular value. So, we get 25 singular values, which are features for every input image. This is done for all 36 images.
5. Feature Set generated. Dimension: 25*36
6. Feature Averaging, reduce it to 15D as some values are close enough to average and store.
7. Feature Reduction, using PCA to make it 3 dimensional.
8. Training completed, follow the steps again for test images as well and predict.

3. RESULTS

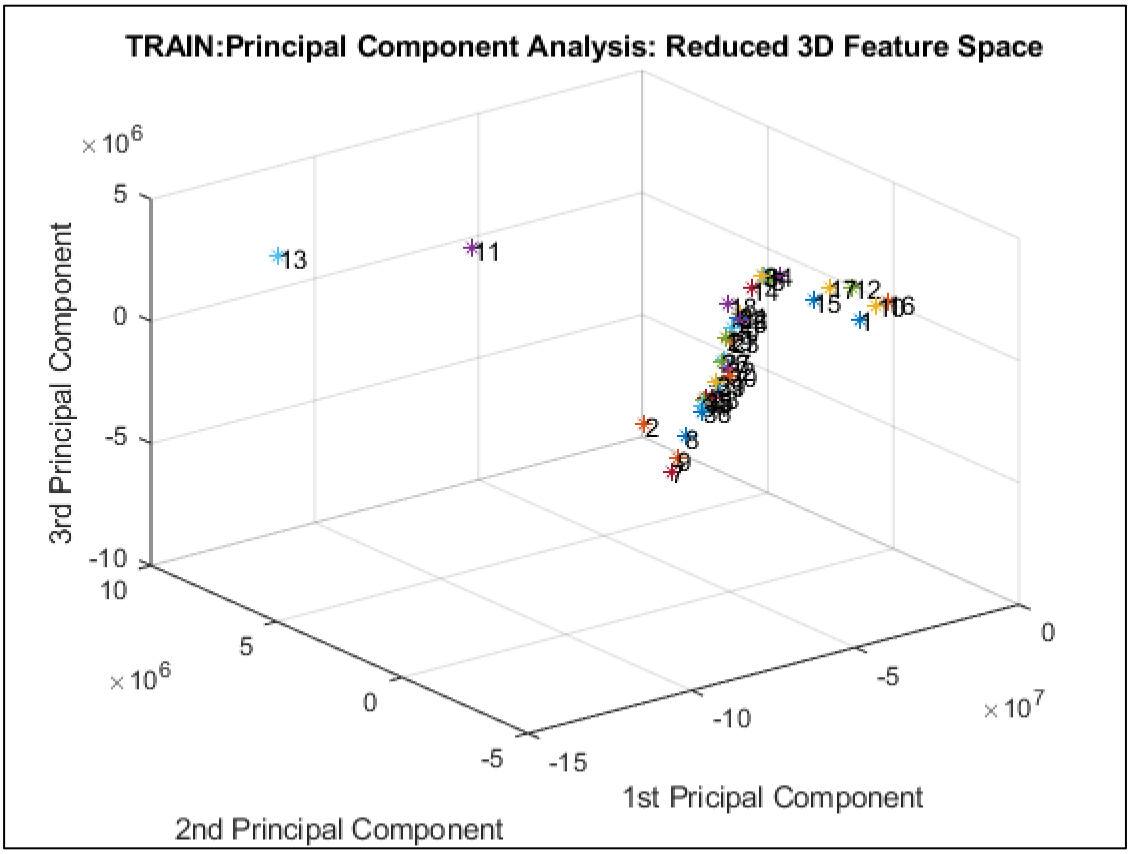


Fig. 1.1: Train Plot PCA

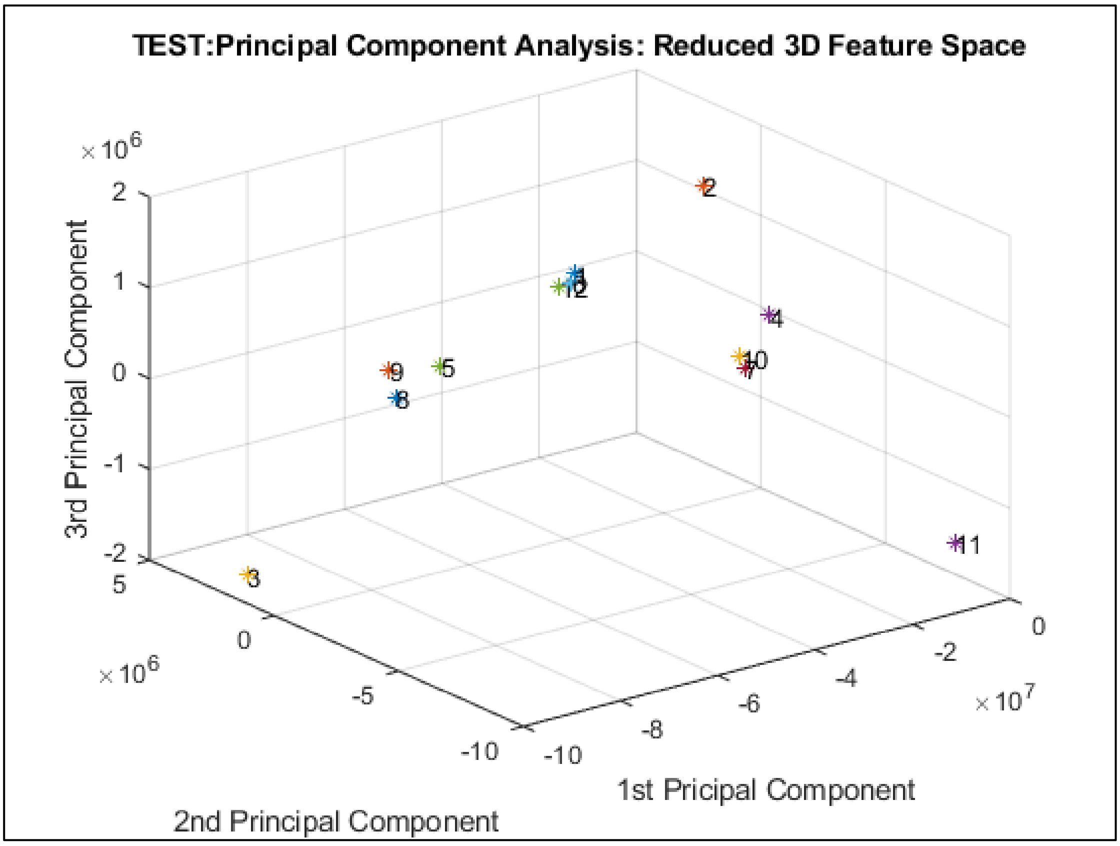


Fig. 1.2: Test Plot PCA

The 3D plot of test samples shows that data points are formed in cluster. These groups are as follows:
Group 1: 2,3,11
Group 2: 4,7,10
Group 3: 1,6,12
Group 1: 5,8,9.

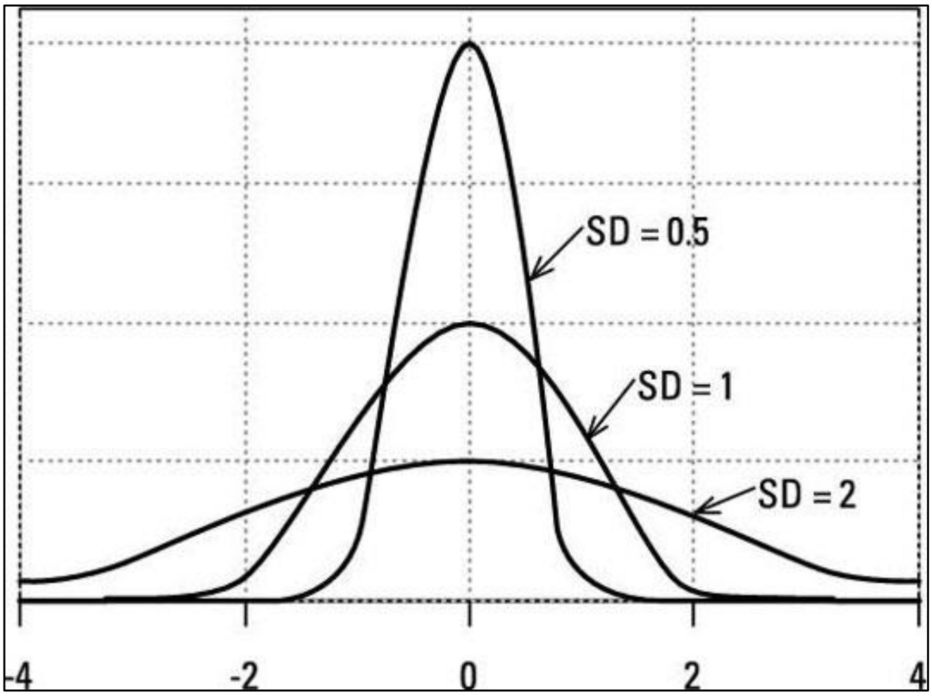
So, we can conclude that these group belong to same category. By visual inspection of testing data set, we confirmed that the predicted labels are correct.

4. DISCUSSION
   a. Feature Extraction
      The 25 filters are applied to 36 input images and feature matrix is created.

   b. Discriminant Power:

LAW FILTERS	VARIANCE
L5*L5’	9.34E+14
L5*E5’	2.29E+12
L5*S5’	4.71E+11
L5*W5’	5.05E+11
L5*R5’	4.43E+12
E5*L5’	1.75E+12
E5*E5’	5.08E+09
E5*S5’	1.17E+09
E5*W5’	2.82E+09
E5*R5’	1.84E+10
S5*L5’	5.82E+11
S5*E5’	9.99E+08
S5*S5’	1.36E+09
S5*W5’	5.05E+09
S5*R5’	2.67E+10
W5*L5’	1.16E+12
W5*E5’	3.43E+09
W5*S5’	8.98E+09
W5*W5’	3.39E+10
W5*R5’	1.66E+11
R5*L5’	1.68E+13
R5*E5’	5.4E+10
R5*S5’	1.38E+11
R5*W5’	5.12E+11
R5*R5’	2.57E+12

The L5*L5’ has the highest variance, so it has weakest discriminant power. The data points are more spread in this feature. The graph of the bell curve is more flattened which approximates to level graph which means clusters are far apart. This makes it the weakest discriminant power.
The S5*E5’ has the lowest variance, so it has largest discriminant power. The data points are more clustered in it. The bell curve of low variance means that data points are closer to each other and thus makes it largest discriminant power.



$$Standard\ \ Deviation(S.D.) = \sqrt{Variance\ (V)}$$

c. The 3D plots of input train images are shown in the result section. We can see that images are grouped into 4 classes. Images 1 to 9 are clustered so as 10 to 17, 18 to 27 and 28 to 36.
The PCA is implemented for the test data and we can see classification of images. The accuracy isn't great and so we get some error which is misclassified.

b) <u>Advanced Texture Classification - Classifier Explore</u>

1. ABSTRACT AND MOTIVATION
As discussed in above section the importance of texture classification remains same. Moving forward, we need to classify these images into correct labels. We do have unsupervised and supervised learning classifier for the same.

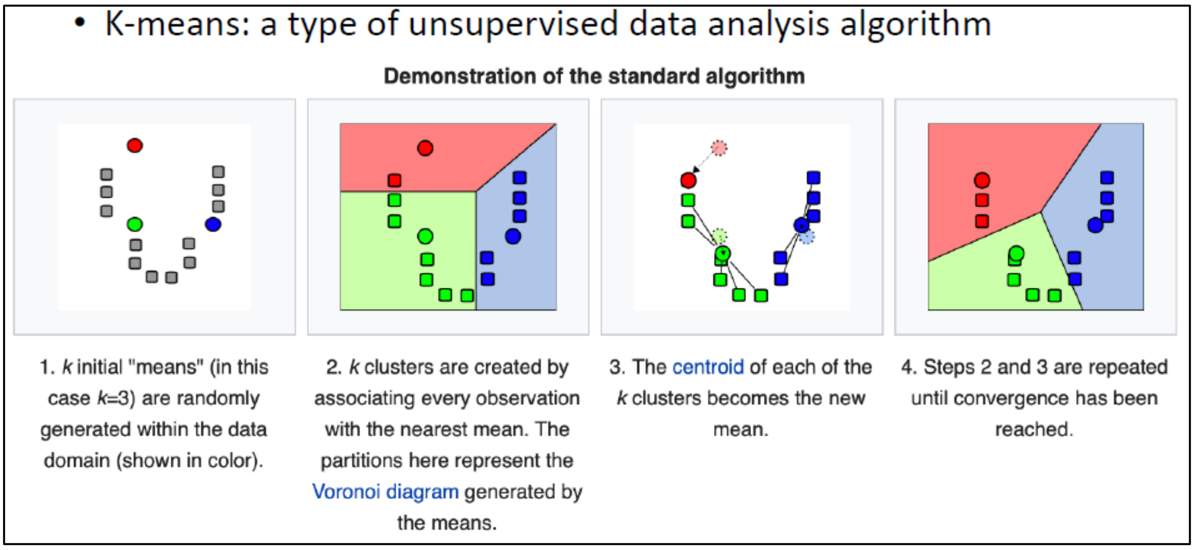
2. APPROACH
Classification of image into correct class is as important as extracting the texture features from a set of input images.
There are 2 classifiers:
Unsupervised Learning and Supervised Learning.

In unsupervised learning, there is no ground truths. In this algorithm, we draw inferences without using labelled responses while in supervised learning, we do have labelled data set and we make efforts to make input close to output label.

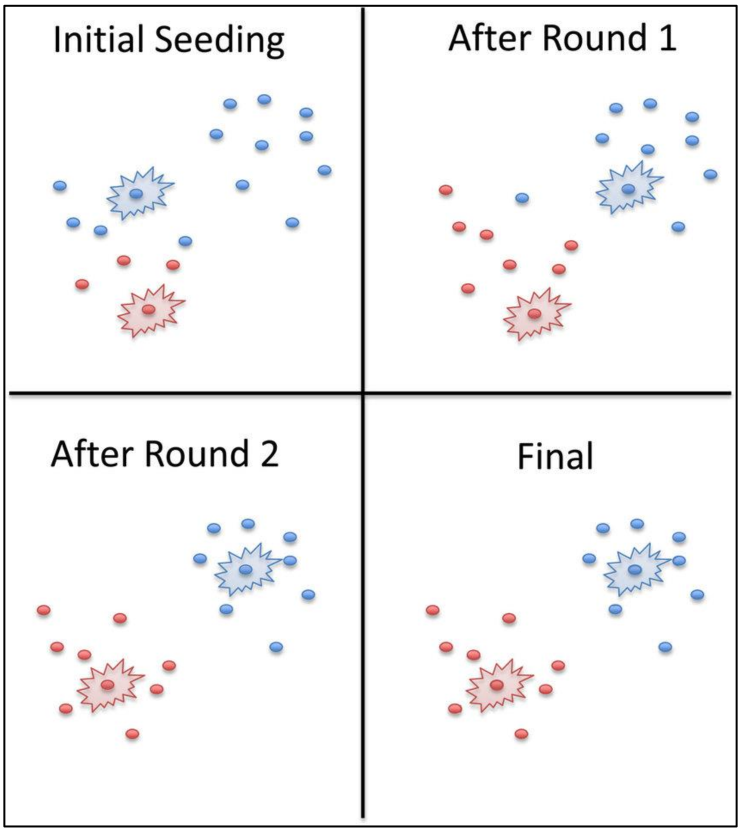
K-means clustering is a type of unsupervised learning that we are going to implement for unsupervised classifier, and we will be using Random Forest (RF) and Support Vector Machine (SVM) for supervised learning classifier. Both results will be compared, and error rate are noted.

MATLAB does have k-means, random forest and SVM inbuilt functions which will be used for the implementation. K-means is applied to 15D and 3D feature space.

K-means:



K-means is a type of unsupervised learning which is used for clustering of data. The k-means model can be trained and used to predict the classes, here labels of the image. Each cluster represents one class. The process is repeated until convergence. As the k-means algorithm is unsupervised one, the output varies each time you run. This is because it takes random initialization for the centroid of the clusters. The random initialization can be points very close to each other. To avoid this, k-means++ is used. It uses initial points which are most far apart which performs better than k-means.

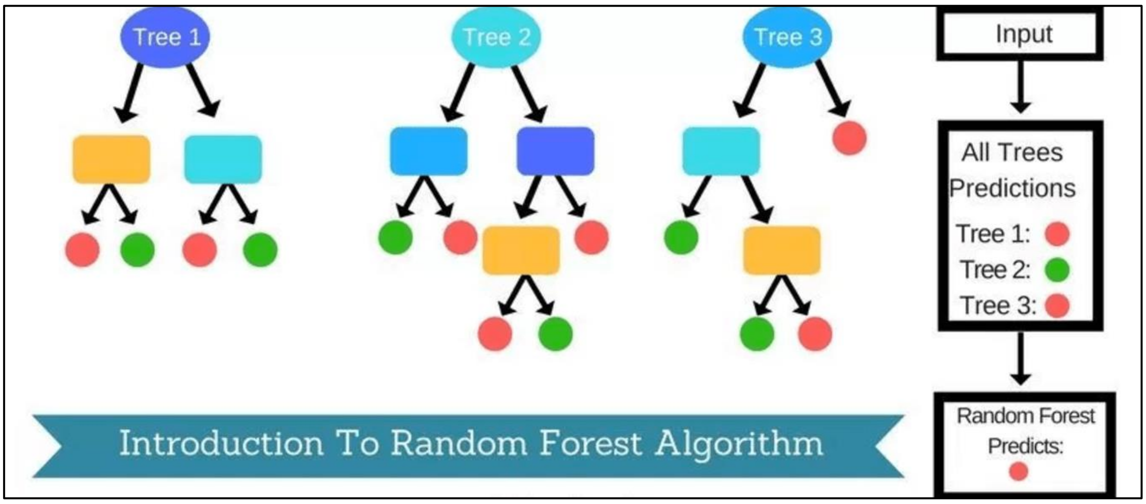


Supervised Learning:

- Random Forest
- Support Vector Machine (SVM)

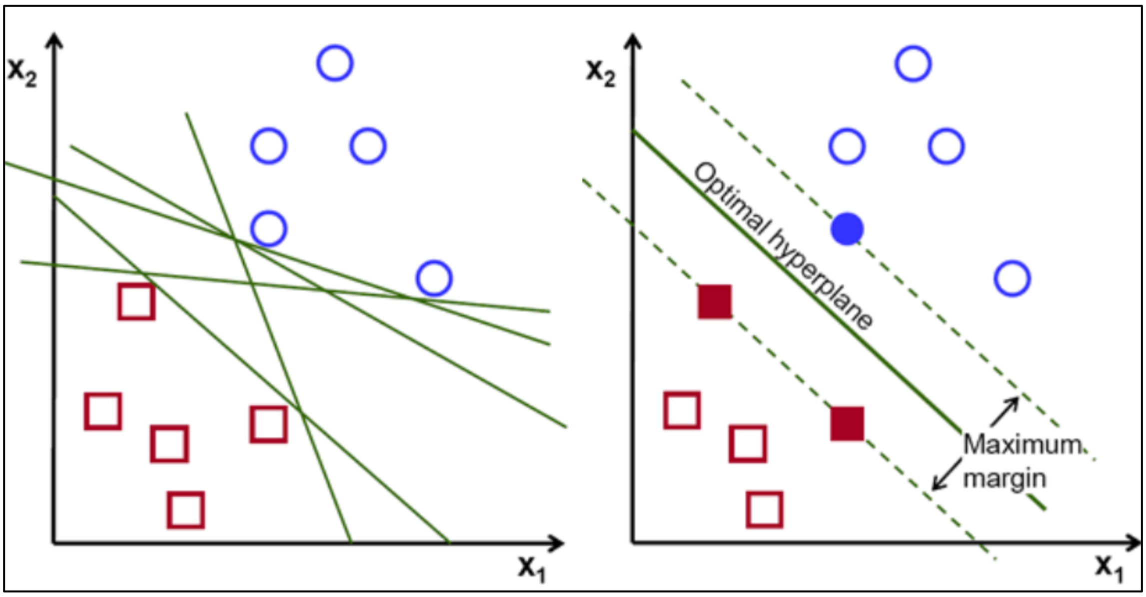
1. First, Label train images by yourself:
   - Blanket: 0
   - Brick: 1
   - Grass: 2
   - Rice: 3
2. Train the two classifiers using the train images and labels
3. Predict labels for test images and calculate error rate

Random Forest:



Random Forest is a type of supervised learning which is used for classification, regression. In this algorithm, a decision tree is made as shown in above figure. These are used for the classification for the prediction of the labels for the test images.

Support Vector Machine:



Support Vector Machine (SVM) is also a type of supervised learning which gives optimal solution and it is better than RF. As shown in the above figure, there are many linear boundaries. But the task is to find out the best fitting line and SVM algorithm does that. SVM maximizes the margin around separating hyperplane.

3. RESULTS

For 3 feature space:

Training:

>> error

error = 30.5556

>> eff

eff = 69.4444

Class Labels Train	Predicted Label Train
3	3
3	3
3	3
3	3
3	3
3	3
3	2
3	2
3	2
2	3
2	2
2	3
2	2
2	1
2	3
2	3
2	3
2	1
1	1
1	1
1	1
1	1
1	1
1	1
1	1
1	1
1	1
4	4
4	4
4	4
4	4
4	1
4	4
4	4
4	4
4	4

For 3 feature space:
Testing:
>> error
error = 16.6667

>> eff
eff = 83.3333

Class Labels Test	Predicted Label Test
1	1
3	2
3	3
2	2
4	4
1	1
2	2
4	4
4	4
2	2
3	2
1	1

For 15 feature space:
Training:
>> error
error = 36.1111

>> eff
eff = 63.8889

Class Labels Train	Predicted Label Train
2	2
2	2
2	2
2	2
2	2
2	2
2	1
2	1
2	1
4	2
4	1
4	2
4	4
4	3
4	2
4	2
4	2
4	3
3	3
3	3
3	3

3	3
3	3
3	3
3	3
3	3
3	3
1	1
1	1
1	3
1	1
1	3
1	1
1	1
1	1
1	1

<u>For 15 feature space:</u>
<u>Testing:</u>
>> error
error = 33.3333

>> eff
eff = 66.6667

Class Labels Test	Predicted Label Test
4	4
2	2
2	3
3	2
1	1
4	4
3	2
1	1
1	1
3	2
2	2
4	4

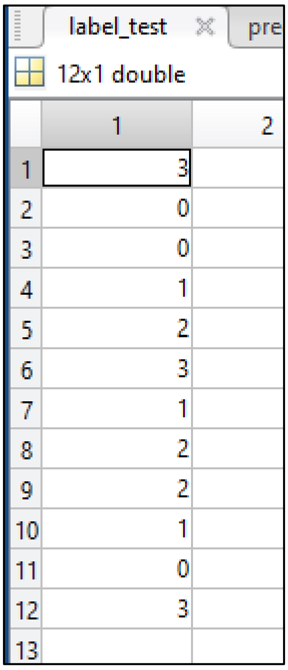
Supervised:

1. Random Forest (RF)

   >> error
   error =
   16.6667

   >> eff
   eff =
   83.3333

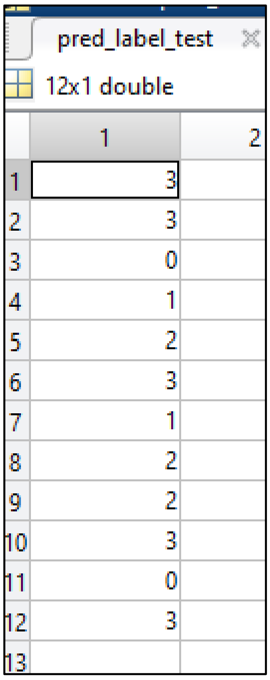
   There are 2 misclassified points.



label_test

12x1 double

	1	2
1	3	
2	0	
3	0	
4	1	
5	2	
6	3	
7	1	
8	2	
9	2	
10	1	
11	0	
12	3	
13		



pred_label_test

12x1 double

	1	2
1	3	
2	3	
3	0	
4	1	
5	2	
6	3	
7	1	
8	2	
9	2	
10	3	
11	0	
12	3	
13		

2. Support Vector Machine (SVM)

>> error
error =
8.3333

>> eff
eff =
91.6667

There is one misclassified point in SVM

label_test
12x1 double

	1	2
1	3	
2	0	
3	0	
4	1	
5	2	
6	3	
7	1	
8	2	
9	2	
10	1	
11	0	
12	3	
13		

label_test3
12x1 double

	1	2
1	3	
2	0	
3	1	
4	1	
5	2	
6	3	
7	1	
8	2	
9	2	
10	1	
11	0	
12	3	
13		

4. DISCUSSION
   a. The k-means algorithm is applied to 15D and 3D feature space. The results are compared visually of 15D and 3D space.

      K-means 15D and 3D comparison:

      As the dimension is higher, the feature space is sparsely populated which leads to more misclassification (centroids are far away in higher dimensions). Here, when k-means is applied to 15D there are more misclassification. Now, when k-means is applied to 3D space, the data points are classified and are clustered (centroids are nearer), and hence less misclassification. For conclusion, k-means is better in 3D than 15D.

      Also, comparison is made between Feature Reduction method (PCA) and K-means. It is concluded that Feature Reduction method is more effective than k-means. There is less misclassification in feature reduction. The reason remains the same, when we reduced the dimensions, we get more clustered data and hence less misclassification.
   b. The 3D feature space is applied in supervised algorithm by 2 methods: Random Forest and SVM. Support Vector Machine (SVM) perform better than RF in case of images. Both are supervised learning, and the error rate shows SVM is better than RF.

c) Texture Segmentation

1. ABSTRACT AND MOTIVATION

   Texture Segmentation is one of the important applications of texture classification. This is very useful when there is a single composite image with lot of textures, and we need to segment this composite image and classify the textures.

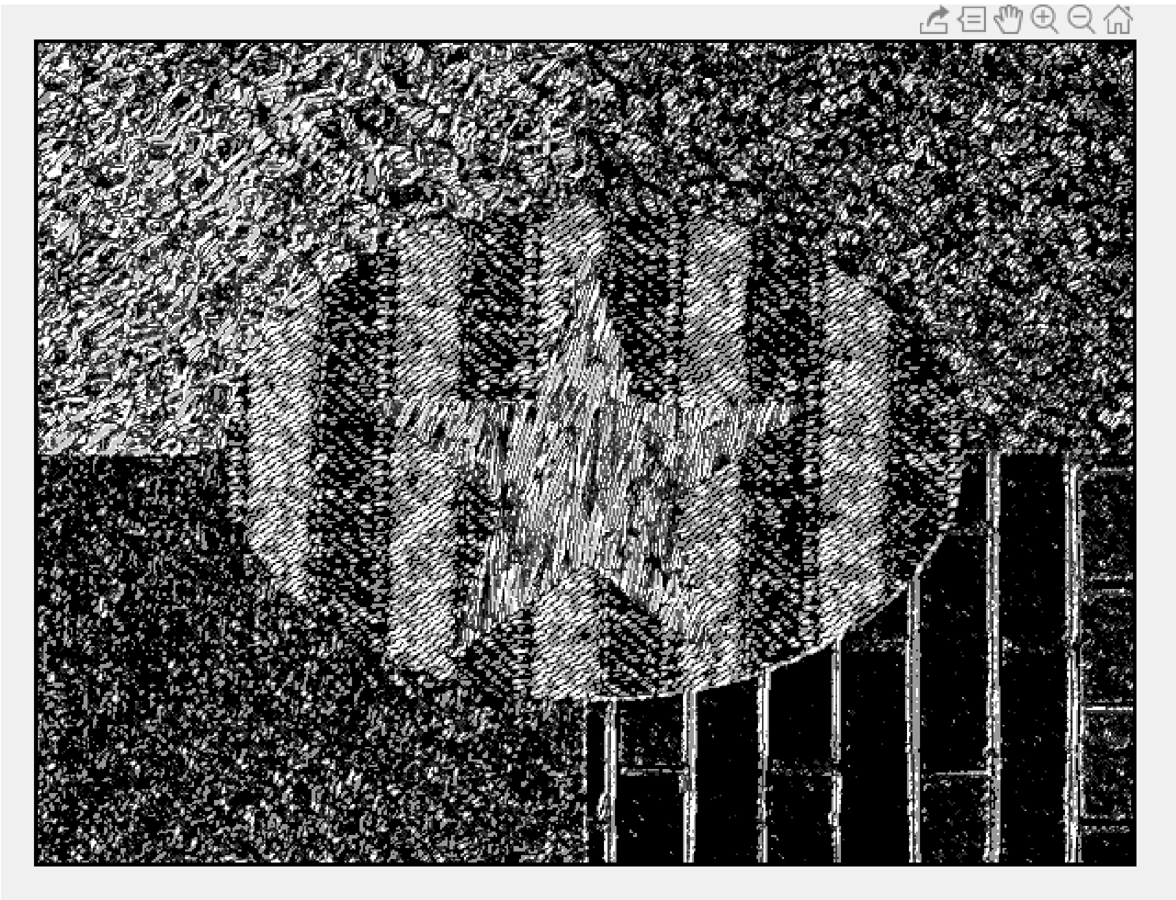
2. APPROACH

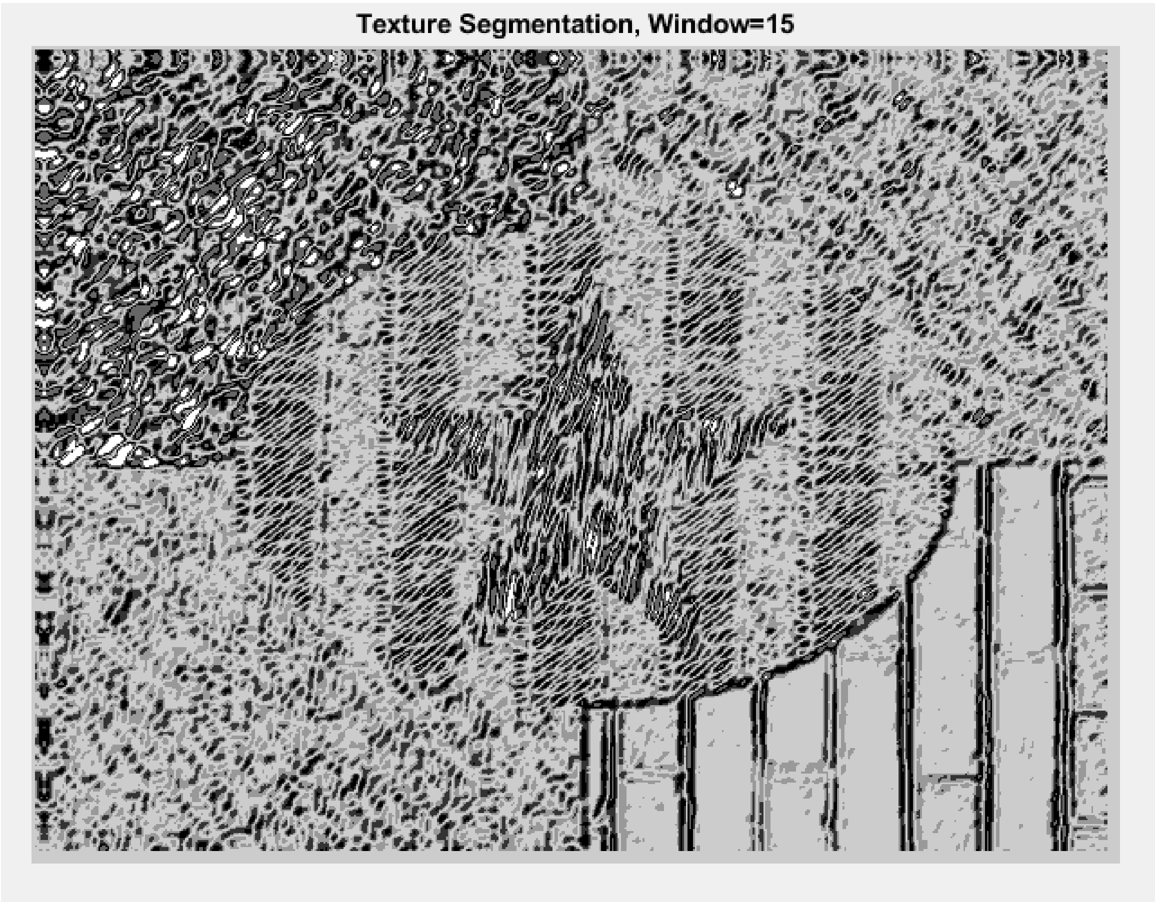
   Texture Segmentation, being a part of texture classification, the part of procedure and algorithm remains the same.
   We do use Law's filter to get 25D feature space for the *'comp.raw'* which is provided to us. These 25 filters must be used for the energy computation which is done by window approach which is different from the method applied earlier.
   The window size makes a huge difference to the texture segmentation over the borders.
   Next, is feature normalization, as seen L5*L5' has least significance it is used for normalization. This value is divided for the 15 features space found above. The resultant is 14D (14*1 matrix). We apply the k-means to 14D matrix, which has 6 classes. We define 6 segments in the range 0 to 255. The range used is 0, 51, 102, 153, 204, 255. Each texture is given one value and the composite image is segmented.

   Algorithm:

   1. Read comp image, apply boundary extension.
   2. Apply 5*5 filters(total 25 filters) on image to get 25 filtered image. (i.e. 25 gray level images)
   3. Making it to 15 filtered image. (i.e. 15 grey level images).
   4. Applying appropriate window to calculate the average of the matrix.
   5. The new matrices has all the pixels with the average value to get 15 dimensional feature space.
   6. Feature Normalization, as L5*L5' is not useful so it is divided for all the feature space.
   7. New Feature space: 14*1.
   8. K-means algorithm with 6 labels.
   9. Range used is 0, 51, 102, 153, 204, 255.
   10. New image generated and displayed with each texture has different grey level.
   11. Vary window size to adjust for a better result.

3. RESULTS




Texture Segmentation, Window=15

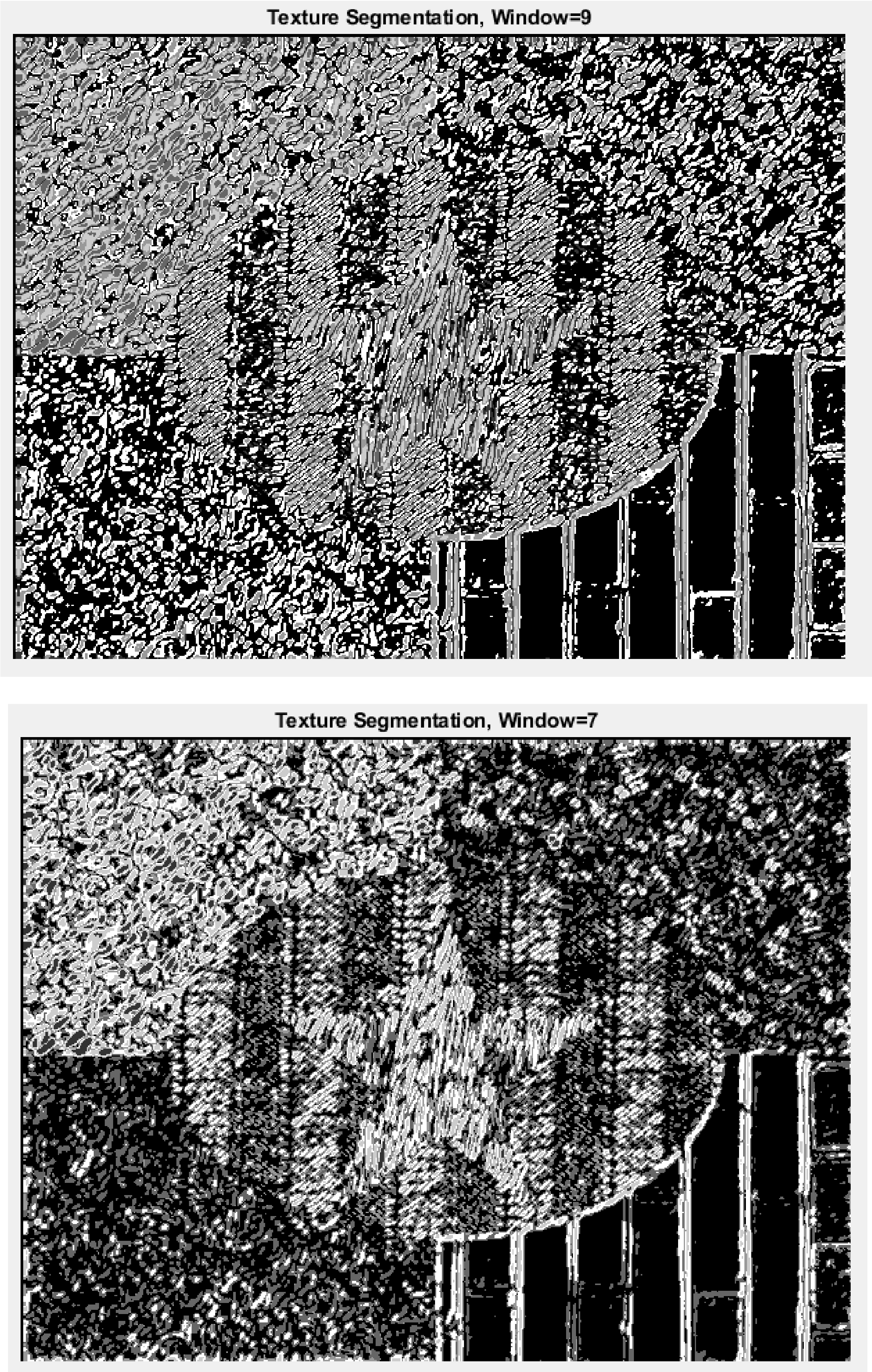


Fig. 2.3: Texture Segmentation images with different window size

4. DISCUSSION
   a. When the window size is increased:
      Boundary in image is more visible
      More holes/gaps can be seen inside the texture image
   b. The widow size =7,9, 15 are shown in the result section. Significant increase in number of artifacts can be seen in the image.
   c. The result is the grey image with each texture given a specific grey level. The result is not the best, but there are ways to improve the result.

d) Advanced Texture Segmentation

1. ABSTRACT AND MOTIVATION

   Texture Segmentation is one of the important applications of texture classification. This is very useful when there is a single composite image with lot of textures, and we need to segment this composite image and classify the textures.

   The results in the previous section wasn't up to mark and the advanced techniques are implemented to improve the same.

2. APPROACH

   There are many ways to improve the results. Some of them are as follows:
   a. Using PCA for feature reduction.
   b. Post processing technique to remove the artifacts/ holes.
   c. Enhancing boundary of 2 adjacent regions by adjusting texture properties.

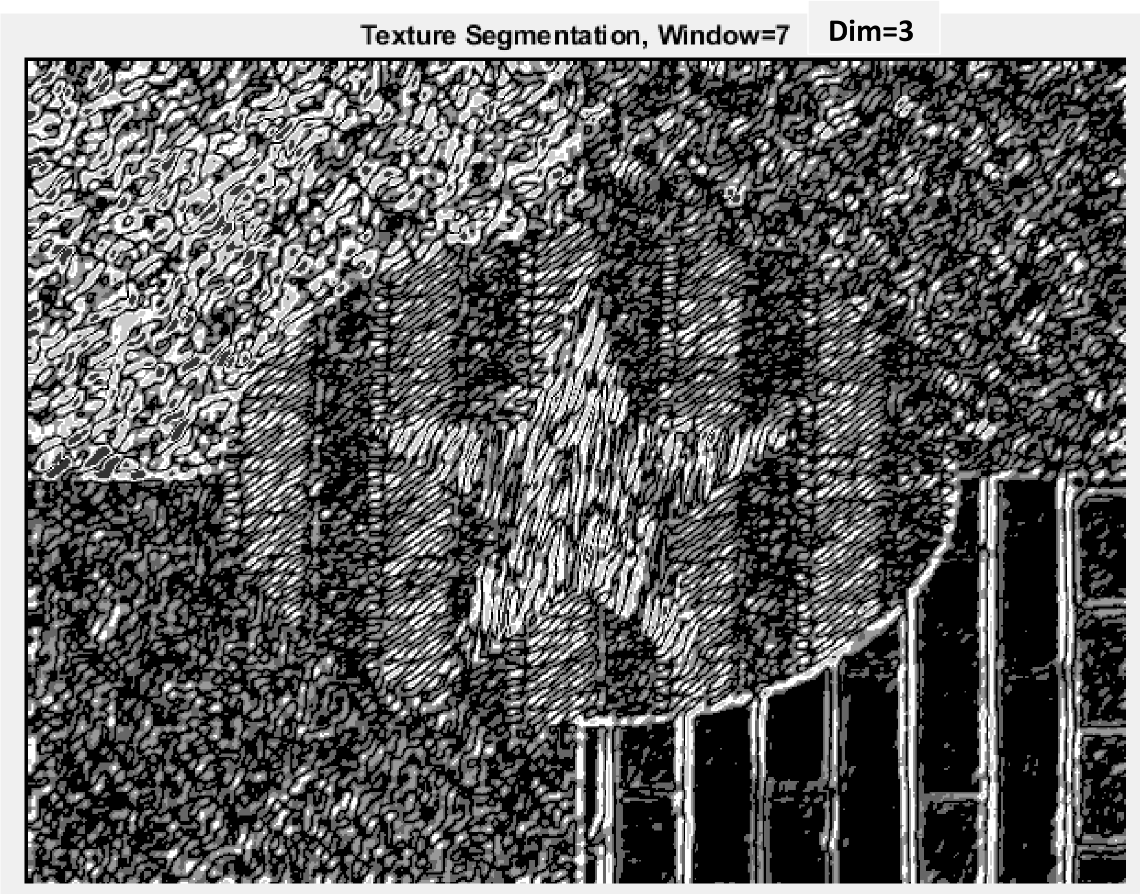
   I am using PCA for feature reduction and then using the reduced matrix to do the segmentation.

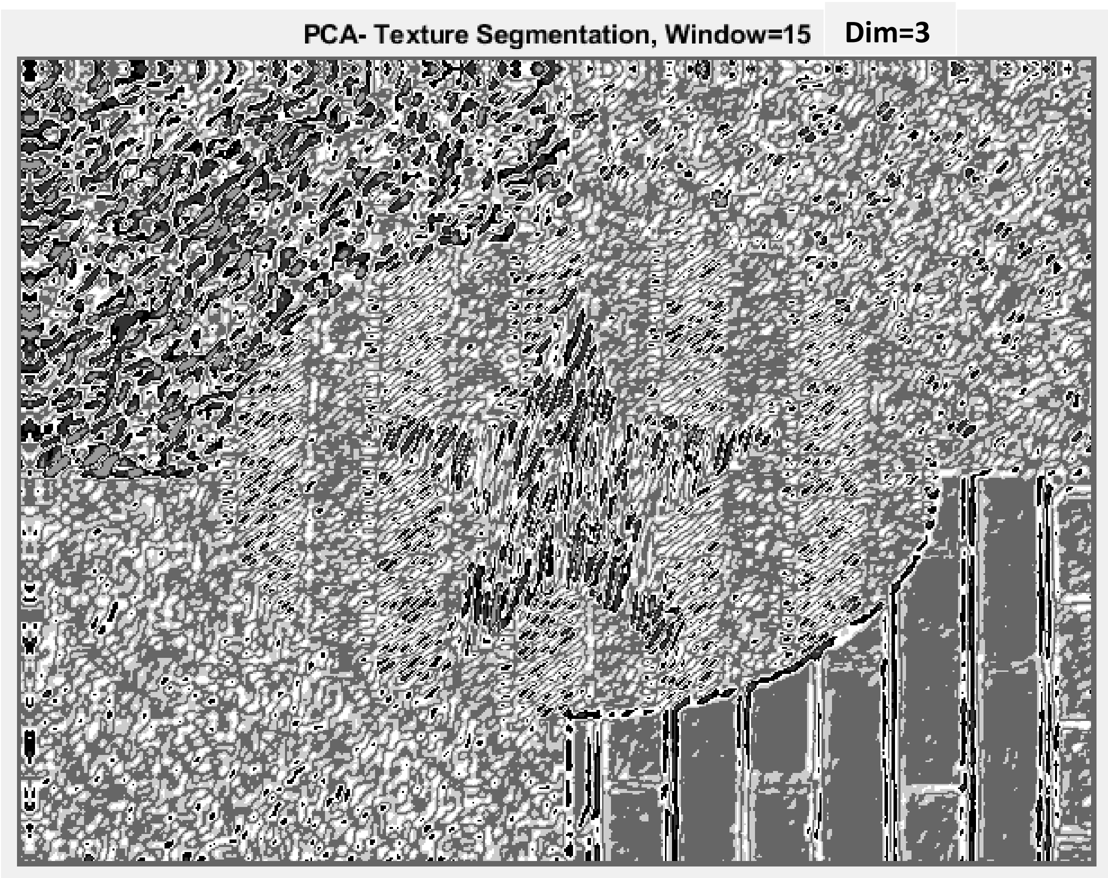
   Reasoning:

   As discussed in the above section, the reduced dimensions are better as the data points are more clustered in it. More the data closely related, it is better to classify which leads to only small chunk of misclassification.

   The PCA function is self-implemented in the above section and used again for doing the same.

3. RESULTS





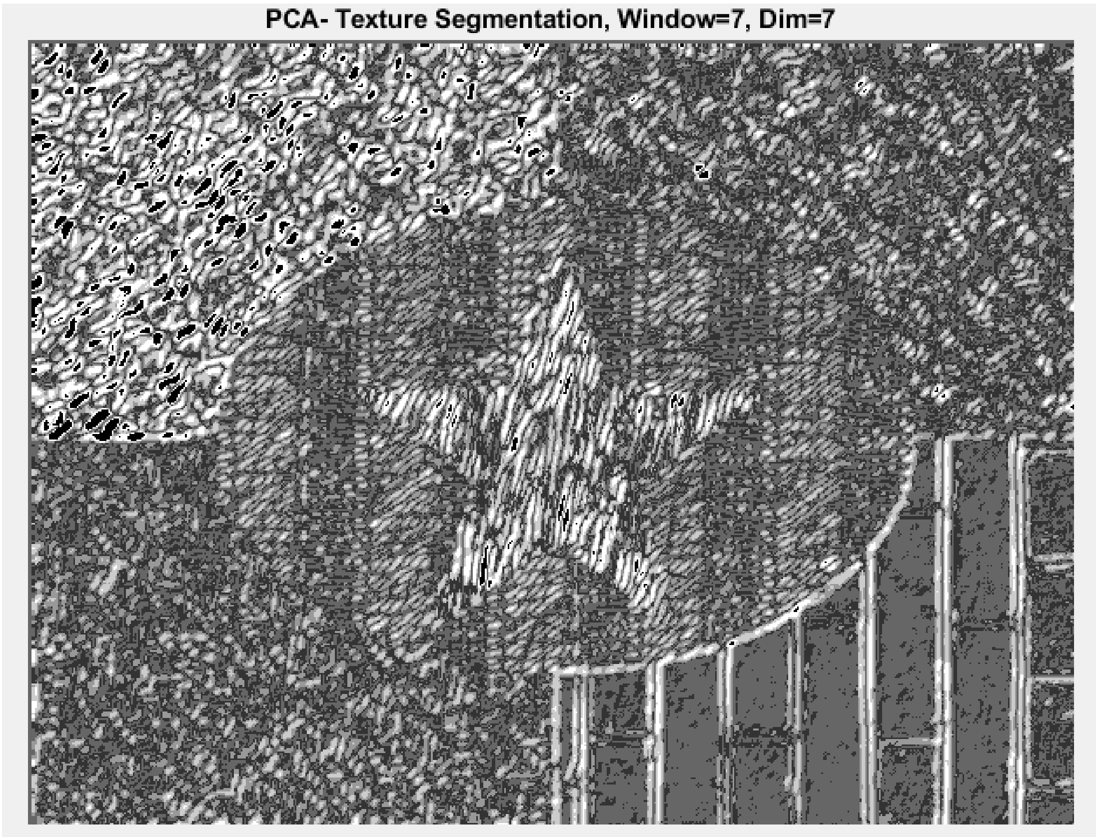
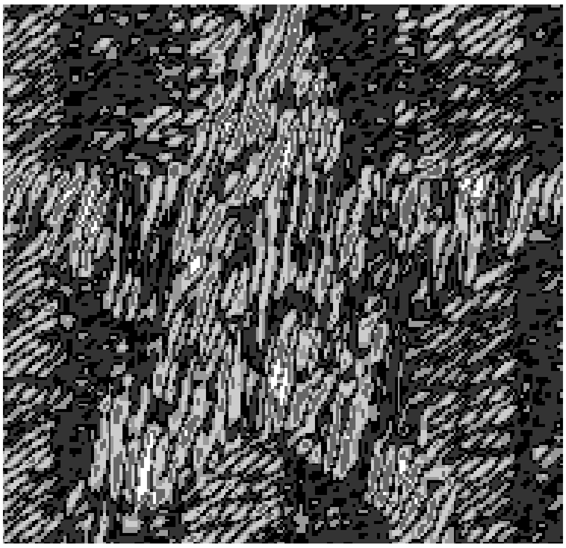


Fig. 2.3: Texture Segmentation images with different window & dimensions with PCA

4. DISCUSSION

This section has been implemented using feature reduction by Principal Component Analysis (PCA).

a. PCA improves the performance significantly as it is using reduced feature space.
b. It is less computationally expensive compared to k-means.
c. The time required to deal with reduced dimensions is lesser, making it faster than k-means.
d. The results are better when reduced to dimensions 7 to 9.
e. The window size is chosen to be 7 as well.
f. We can see that holes in 14D image are more than obtained in reduced feature vector.
g. Choosing dimensions is tricky, very low and very high dimensions can distort the image.



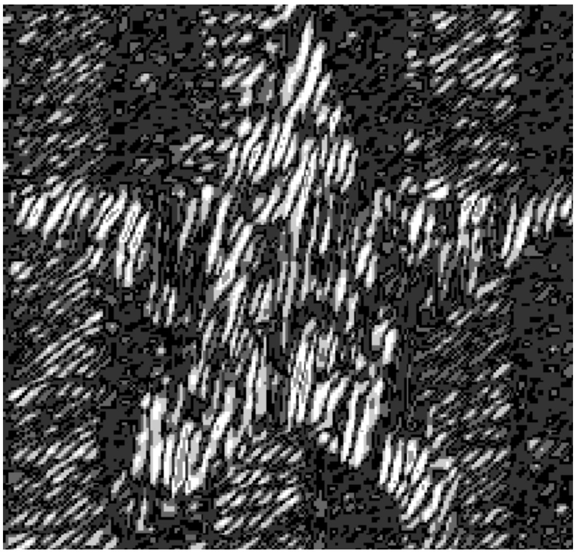


Fig. 2.3: Texture Segmentation image comparison
Left image- Kmeans on 14D
Right image- Kmeans on 7D (PCA)

Problem 2: Image Feature Extractors

a) Salient Point Descriptor

1. ABSTRACT AND MOTIVATION

   Scale Invariant Feature Transform (SIFT) is very effective in calculating the salient Point Descriptor which is required in many applications. The distinct features of image are extracted by this algorithm. It uses convolutional approach for the same. SIFT works very well and has lots of advantages over other approaches and so is preferred all day along to detect features.

2. APPROACH

   The following steps are used to generate image features. The method has been called Scale Invariant Feature Transform (SIFT).
   1. Scale-space extrema detection
   2. Key point localization
   3. Orientation assignment
   4. Keypoint descriptor

3. ANSWERS

   The answers are with reference after reading the paper given.
   1. SIFT is robust to geometric modifications like scaling, rotation, translation. Also, it is robust to affine transforms, addition of noise and illumination effects.

   2. A. Robustness to Scaling:

      The robustness to scaling is achieved by Scale space extrema detection. The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.

      B. Robustness to Rotation:

      It is achieved by assigning a consistent orientation to each key point based on local image properties. The key point descriptor can be represented relative to this orientation and therefore achieve invariance to image rotation.

      C. Robustness to Affine Transforms:

      The Hough transform is used to identify all clusters with at least 3 entries in a bin. Each such cluster is then subject to a geometric verification procedure in which a least-squares solution is performed for the best affine projection parameters relating the training image to the new image.

   3. The robustness of illumination is achieved by following:
      a. The feature vector is normalized to unit length. Contrast change is to be cancelled by vector normalization which is achieved by multiplying by a constant. A brightness change in which a constant is added to each image pixel will not affect the gradient values, as they are computed from pixel differences. Therefore, the descriptor is invariant to affine changes in illumination.
      b. Reducing the influence of large gradient magnitudes by thresholding the values in the unit feature vector. The value chosen is about 0.2. This means that matching the

magnitudes for large gradients is not significant, and that the distribution of orientations has greater focus which reduces illumination effects.

4. Advantages of SIFT using Difference of Gaussian (DoG) over Laplacian of Gaussian (LoG):
   a. DoG kernel is separable while LoG kernel is not.
   b. The computational time of DoG is low as compared to LoG.
   c. DoG is approximation to LoG

5. The output vector size in SIFT in the original paper is 160.
   a. The descriptor is formed from a vector containing the values of all the orientation histogram entries. The best results are achieved with a 4x4 array of histograms with 8 orientation bins in each. Therefore, the experiments in the paper use a 4x4x8 = 128 element feature vector for each key point.
   b. The same procedure is applied in higher octaves to reduce illumination with 2*2 array, so 2x2x8= 32
   c. Total vector size: 128+32=160.

b) <u>Image Matching</u>

1. ABSTRACT AND MOTIVATION

   Image Matching is one of the best applications using SIFT. The keypoints are generated as described by the paper provided. This method works so well as is invariant to geometrical modifications.

2. APPROACH

   The steps involved in using SIFT for key point extractor as given in the above section. The correct feature point is so important as the task is to match the same point in another image as well.

   Nearest Neighbouring method is also used for the matching between 2 images. It is defined as the minimum Euclidian distance between each of the descriptor of 2 images.

   <u>Algorithm</u>:
   1. Read all images for where you need to match the features.
   2. Taking grey level image and extracting SIFT features for reference image (Here- ‘*Husky3.raw’*)
   3. Take 2nd image which is to be matched and extract SIFT features (here- ‘*Husky1.raw’,* ‘*Husky2.raw’ and* ‘*Puppy1.raw’*).
   4. Finding largest L2 norm for the descriptor for reference image.
   5. Finding Euclidian distance to each feature of another image using L2 norm of reference image.
   6. Search for minimum Euclidian distance.
   7. This yield feature and descriptor match between 2 images.

3. RESULTS

Part A Results:

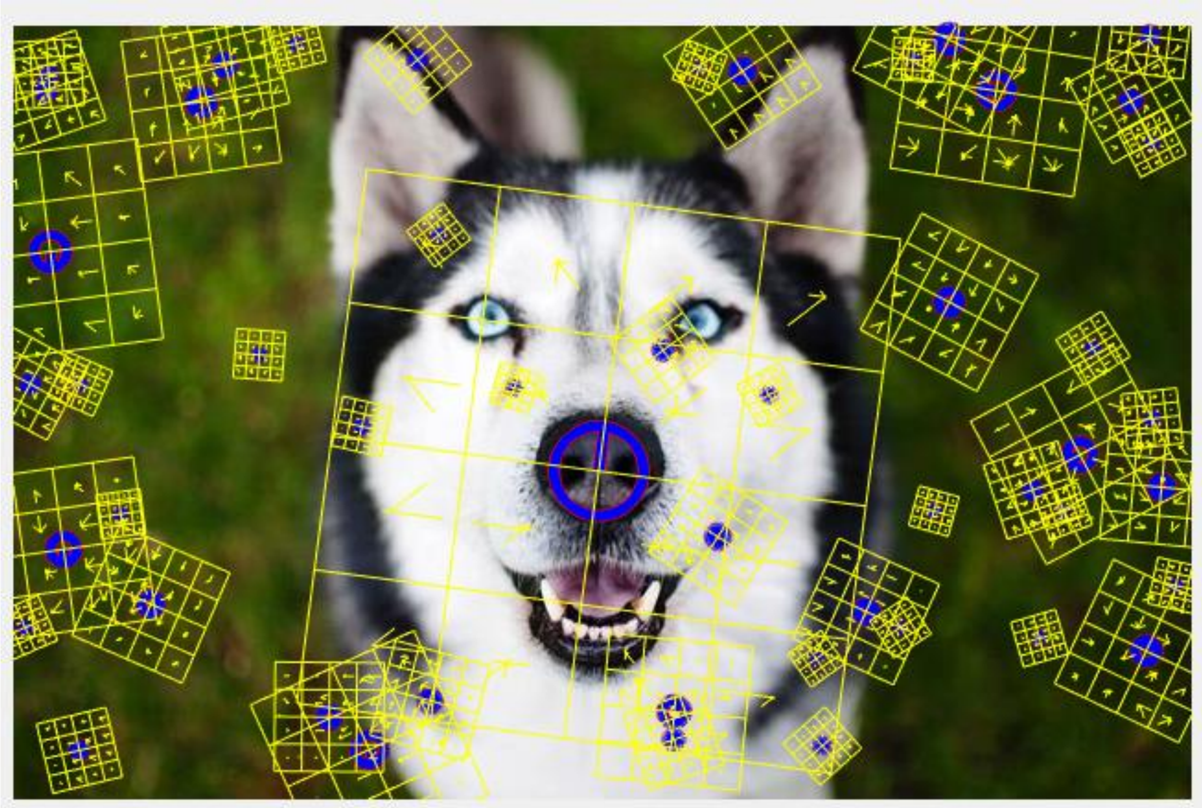


Fig. 2.1: Husky3 SIFT

Fig. 2.2: Husky1 SIFT

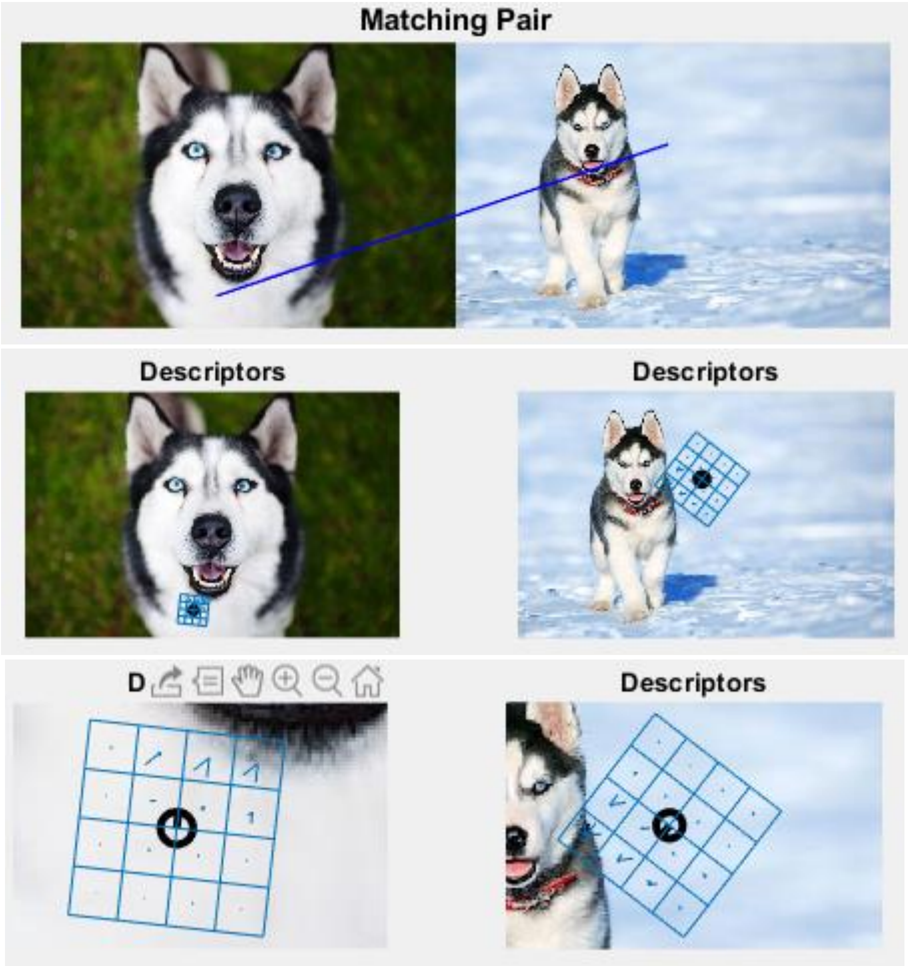
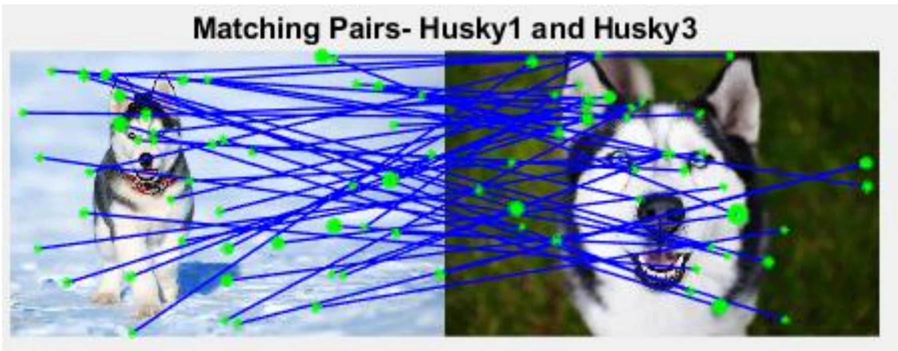


Fig. 2.3: Keypoint Matching and descriptors

Part B Results:



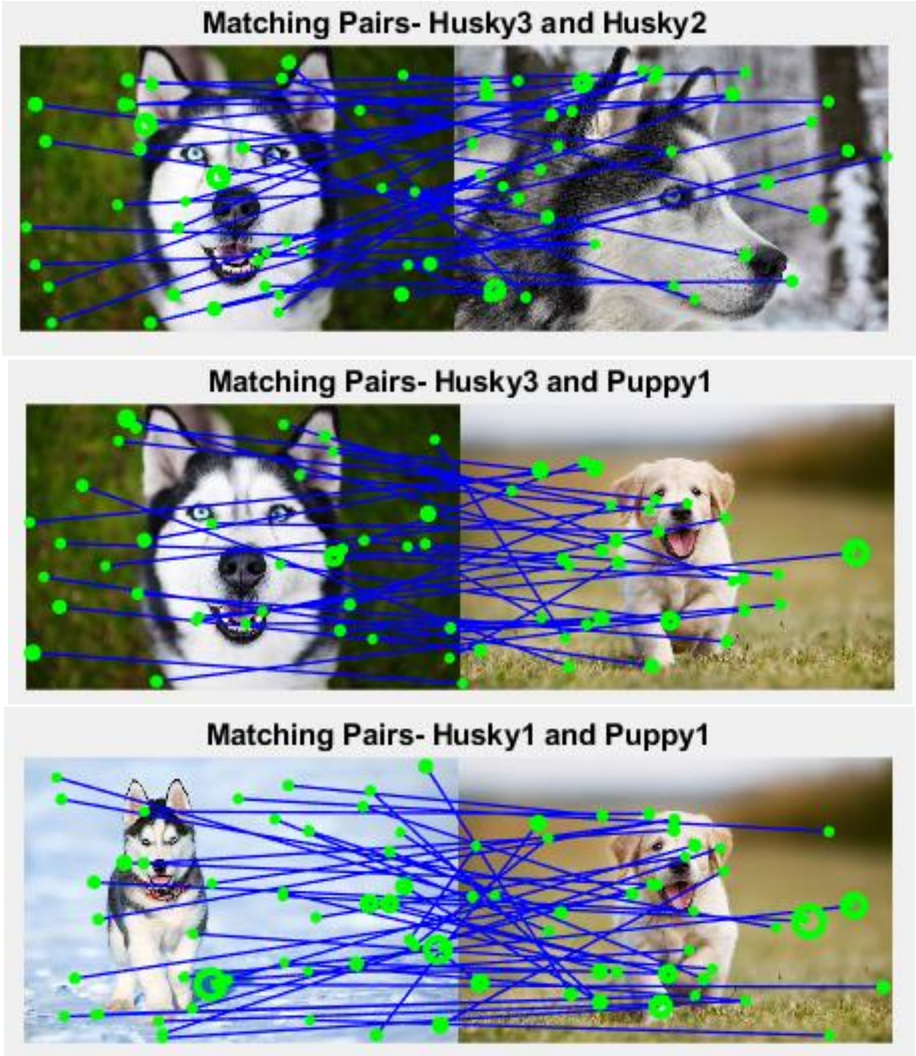


Fig. 2.4: SIFT pairs of different images.

4. DISCUSSION

F = [VL_SIFT](I) computes the SIFT frames (keypoints) F of the image I. I is gray scale single image. F has the format [X;Y;S;TH], where X,Y is the (fractional) center of the frame, S is the scale and TH is the orientation (in radians).

[F,D] = [VL_SIFT](I) computes the SIFT descriptors.. Each column of D is the descriptor of the corresponding frame in F. A descriptor is a 128-dimensional vector of class UINT8.

a. The key points of 'Husky3.raw' and 'Husky1.raw' are matched. The location, scale and orientation of the 2 images are as follows:

Location of Husky3:	**290.978729248047**
	74.7288970947266
Scale:	**2.12735056877136**
Orientation:	**-1.29361146845886**

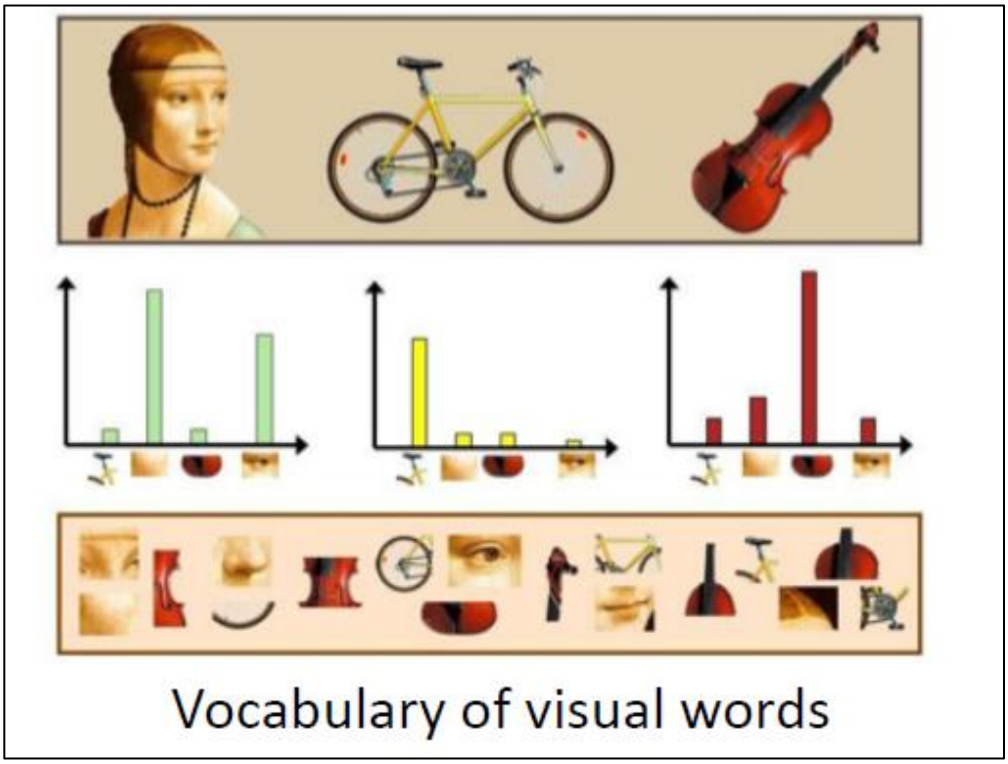
Location of Husky1: **310.378417968750**
273.536560058594
Scale: **1.74113810062408**
Orientation: **-3.96748957010662**

b. The same procedure is applied taking rest of the images and image matching is achieved. The image matching does works and fails for some pairs.
   As seen from the above results, when the image is side view of dog, the matching does not work so well. In some cases, the ear of the reference image is matched to the ground and so on.
   When puppy image is used, it is still a dog but has it different breed, it does not work very well as both dogs look quite different.
   Husky3 and Husky1 are in the same orientation as well as same viewing point, so SIFT matching works on those images.

c) Bag of Words

1. ABSTRACT AND MOTIVATION
   Bag of Words is used in image processing to store image features as words. These features are extracted using the various methods like SIFT, SURF, etc. This becomes a very important classification problem. It is very important in predicting the unknown data and know which class it belongs. Bags of Words is widely used in object recognition and prediction.

2. APPROACH
   Bag of Words stores image features which are extracted from SIFT or SURF. The training image is divided into smaller parts to make this small chunk of word called vocabulary of visual words. Various images are taken, and the dataset of visual words is created and stored in dictionary. Then each image is taken and plotted against the histogram.
   The following diagram shows 3 test images from which various dictionary of feature words is created a.k.a. bag of words.

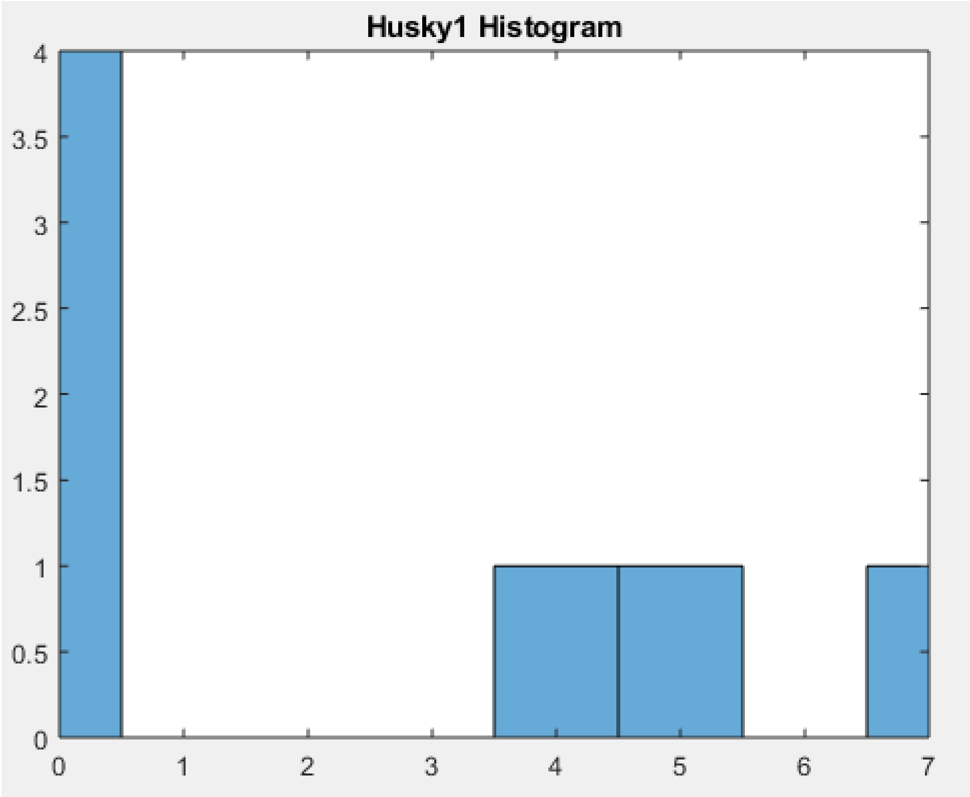


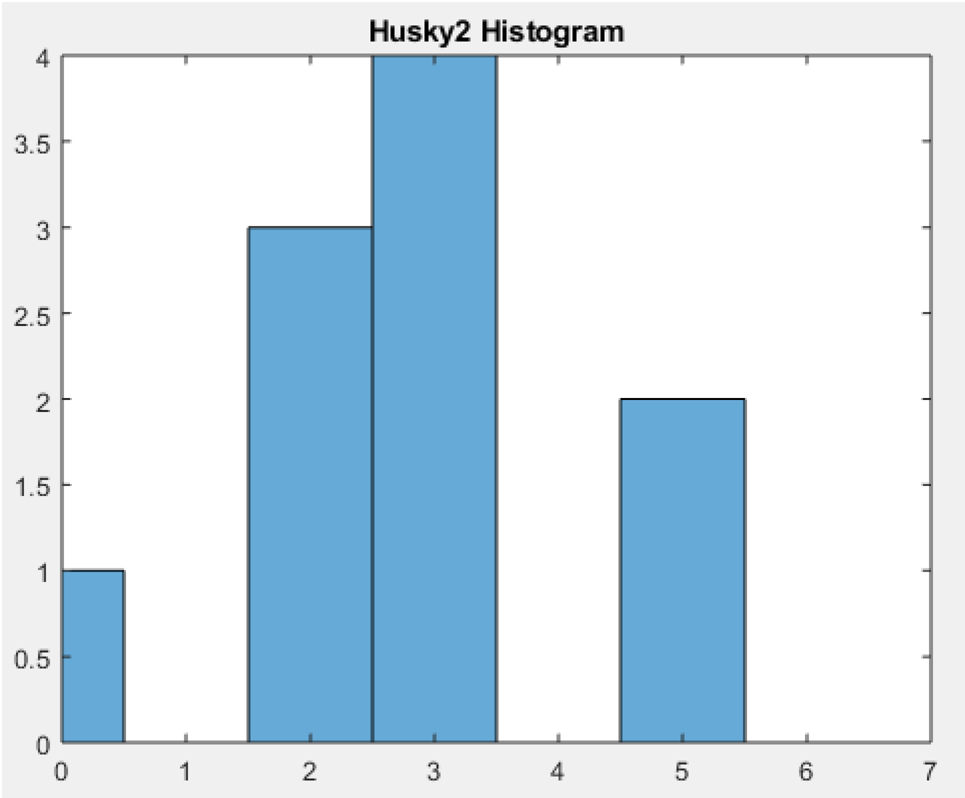
Codebook generation is way of representing the similar features in one group. K-means clustering can be implemented, with its centroid taking as codewords.
The test images features are searched for similar features in codeword and hence prediction of label is made.

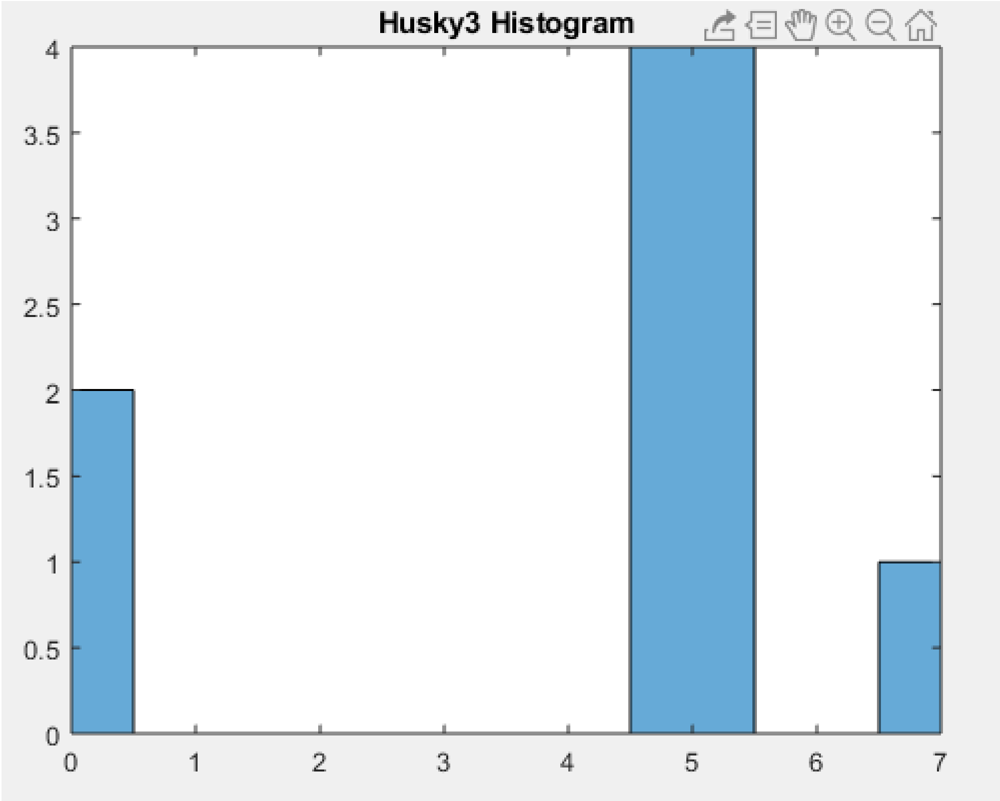
Algorithm:

1. Extract SIFT features of all 4 images.
2. Applying k-means algorithm to form a codebook. This codebook contains 8 bins with each bin denotes the centroid of feature vector that are extracted.
3. Each image is represented as histogram with feature vectors.
4. Codewords for all 4 images.
5. Match 'Husky3.raw' codeword with rest of the 3 images.

3. RESULTS


Husky1 Histogram
4
3.5
3
2.5
2
1.5
1
0.5
0
0 1 2 3 4 5 6 7


Husky2 Histogram
4
3.5
3
2.5
2
1.5
1
0.5
0
0 1 2 3 4 5 6 7



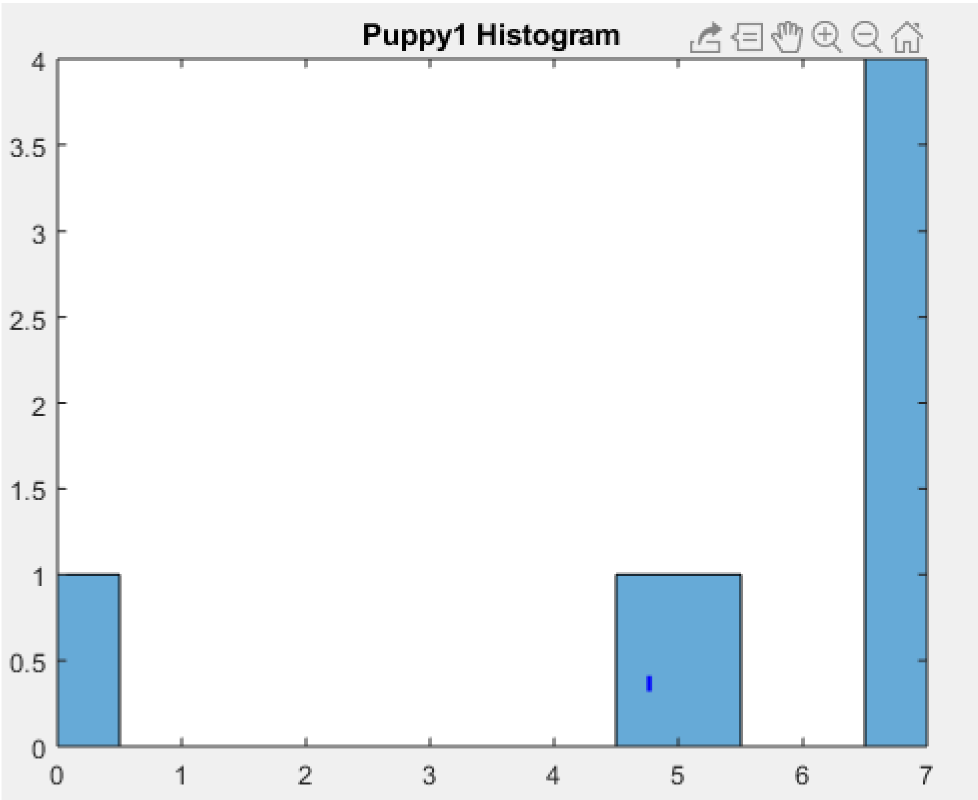


Fig. 2.5: Histograms of all 4 images

4. DISCUSSION

The codeword is generated for all 4 images along with the histograms. The comparison is made between 'Husky3' and rest 3 images.

As by visual inspection of Husky1 and Husky3, we know that face of both Husky's are quite similar and therefore there should be image matching. The result obtained for Histogram of Husky1 shows that there are codewords like Husky3.

For Husky3, the view angle is different, so not all descriptors are match, it is an example of bad matching. The histogram results show the same as well.

For Puppy1, viewing angle is same as that of Husky3, but as we know the dog breeds are different. It's matching also doesn't come out to be good.

REFERENCES:

1. https://www.sciencedirect.com/topics/engineering/texture-classification
2. https://in.mathworks.com/matlabcentral/fileexchange/68943-svd-for-dimension-reduction-in-1d-data
3. https://in.mathworks.com/matlabcentral/answers/435928-how-can-i-use-random-forest-classifier
4. David G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, 60(2), 91-110, 2004.
5. https://dsp.stackexchange.com/questions/37673/what-is-the-difference-between-difference-of-gaussian-laplace-of-gaussian-and
6. https://in.mathworks.com/help/vision/examples/image-category-classification-using-bag-of-features.html