

EE 569 Digital Image Processing

Homework #4

Likitha Lakshminarayanan

Issued: 03/04/2020 Due: 11:59PM, 03/22/2020

Problem 1: Texture Analysis and Segmentation

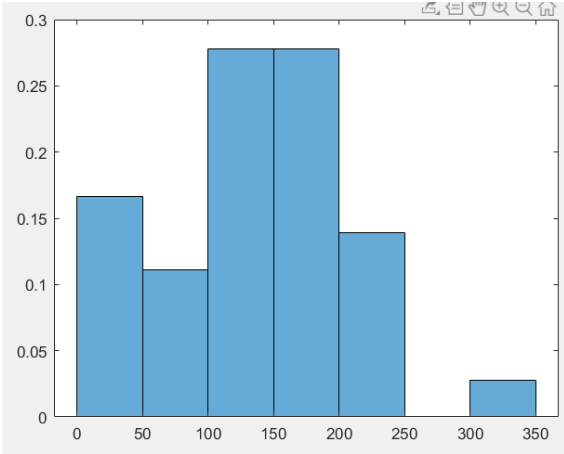
(a)Texture Classification --- Feature Extraction

Procedure:

- The 36 input training images are read. The training image consists of grass, rice, brick and blanket images.
- In order to reduce the illumination effects, the image is subtracted by the mean of the entire image.
- The law filters are of the following form:

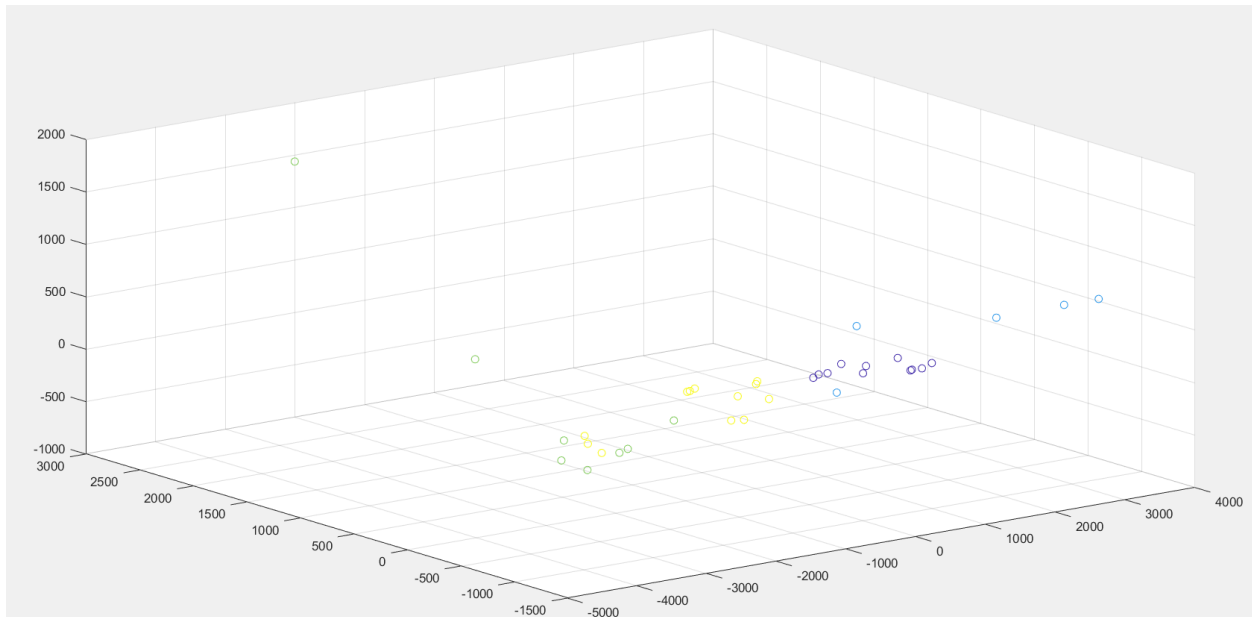
Name	Kernel
L5 (Level)	[1 4 6 4 1]
E5 (Edge)	[-1 -2 0 2 1]
S5 (Spot)	[-1 0 2 0 -1]
W5 (Wave)	[-1 2 0 -2 1]
R5 (Ripple)	[1 -4 6 -4 1]

- The 2D 5x5 law filters are computed by the tensor product between each 1D kernels.
- The image is filtered using the law filters. Thus, producing different textures.
- The energy feature vector is computed which is equal to the absolute sum of the filter output value at each pixel for the entire image. Thus, one image produces a 1x25 vector. It produces a 25D energy vector.
- The 25D energy feature vector is reduced to 15-D energy feature vector by computing the average of every pair. (For example, the L5E5 and E5L5 have similar values).
- Since the training set is 36 images, the final matrix will be 36*15.

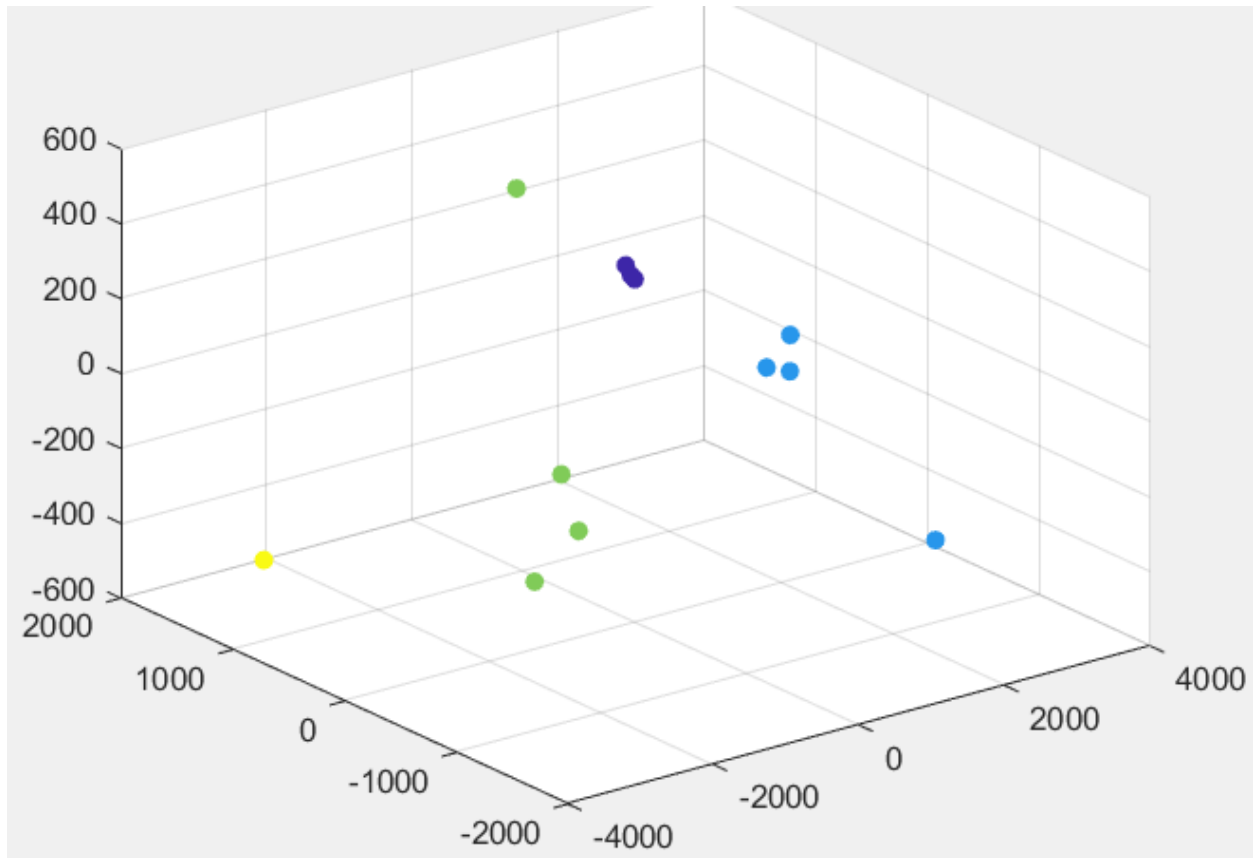


The E5E5 has strong discriminant power

(c) Feature Reduction



PCA (reduced to 3D) followed by kmeans using built in function Training Images

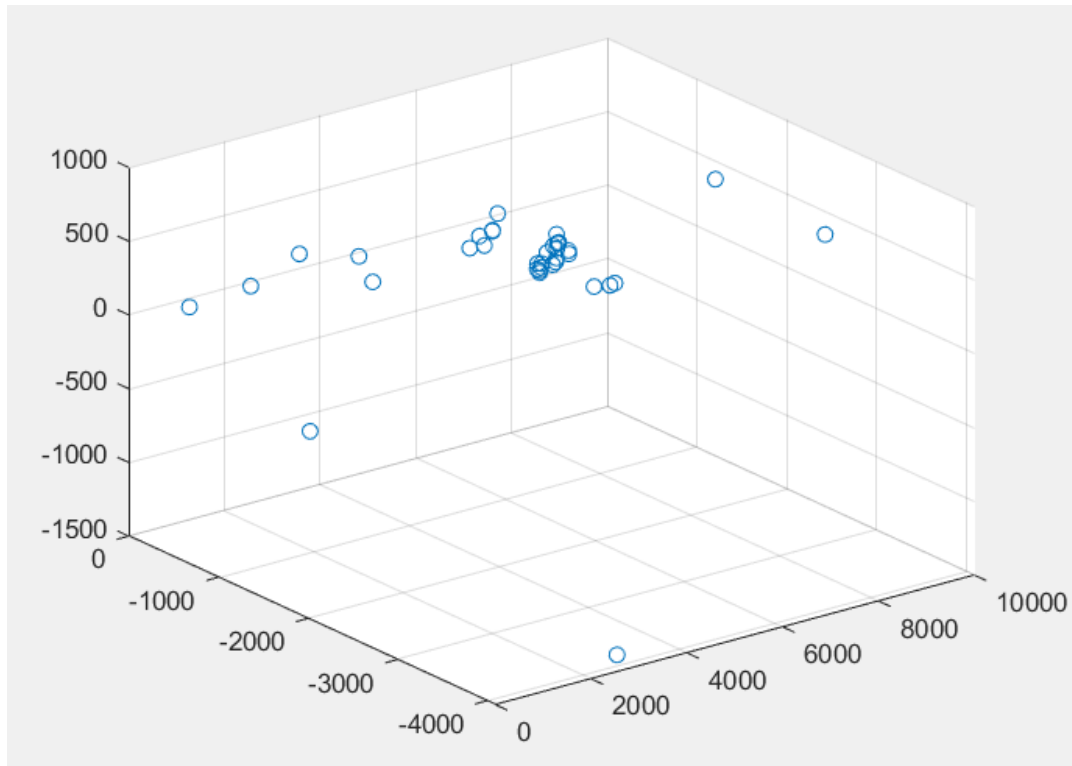


PCA (reduced to 3D) followed by kmeans using built in function Testing Images

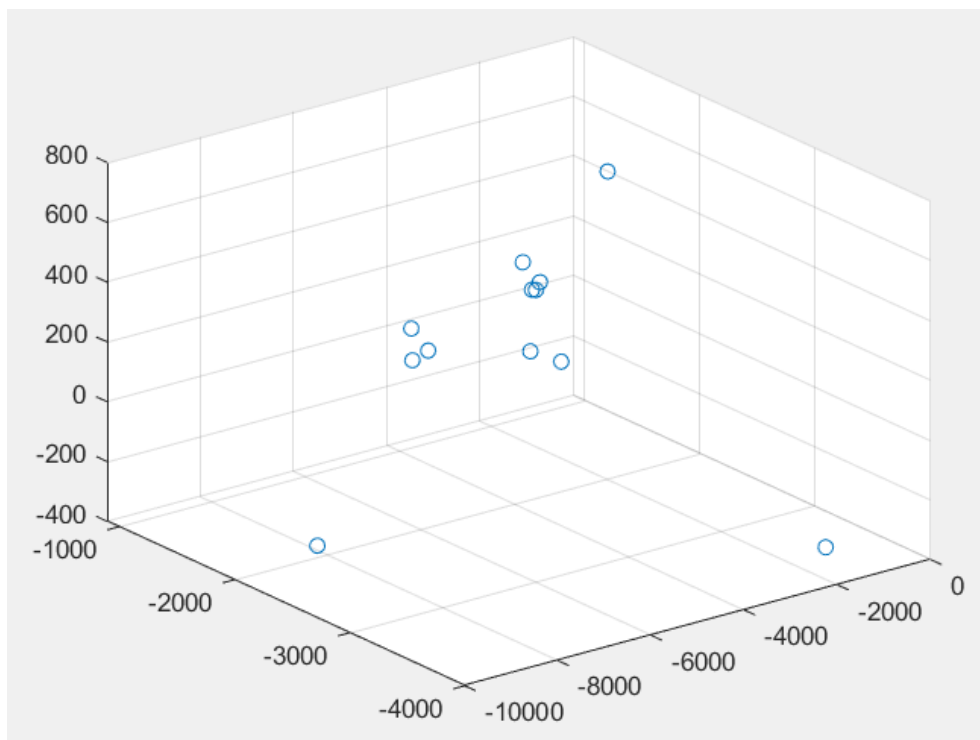
PCA Implemented

Procedure:

- The energy feature vector is computed of size 36x15 for training and 12x15 for testing X
- Compute the mean of the energy feature vector
- Subtract the mean to get zero mean data
- Compute SVD using svds
- The right singular vector with the top three singular values are retained VR.
- The dimension reduced matrix is $YR = X * VR$.



PCA 3D: Training



PCA 3D: Testing Images

(b) Advanced Texture Classification --- Classifier Explore

Test Images:

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

1.Unsupervised

Procedure

- The input training images, and testing images are read.
- The energy feature vectors are produced using the previous method.
- The kmeans algorithm is applied to cluster the energy features of size 12x15.
- The PCA is applied to both the training energy vector and testing energy vector for dimension reduction to 3D.
- The kmeans algorithm is applied to cluster the energy features of size 12x3.
- Thus, 4 clusters for brick, grass, rice and blanket is produced.
- The testing accuracy is calculated for 15D and 3D.

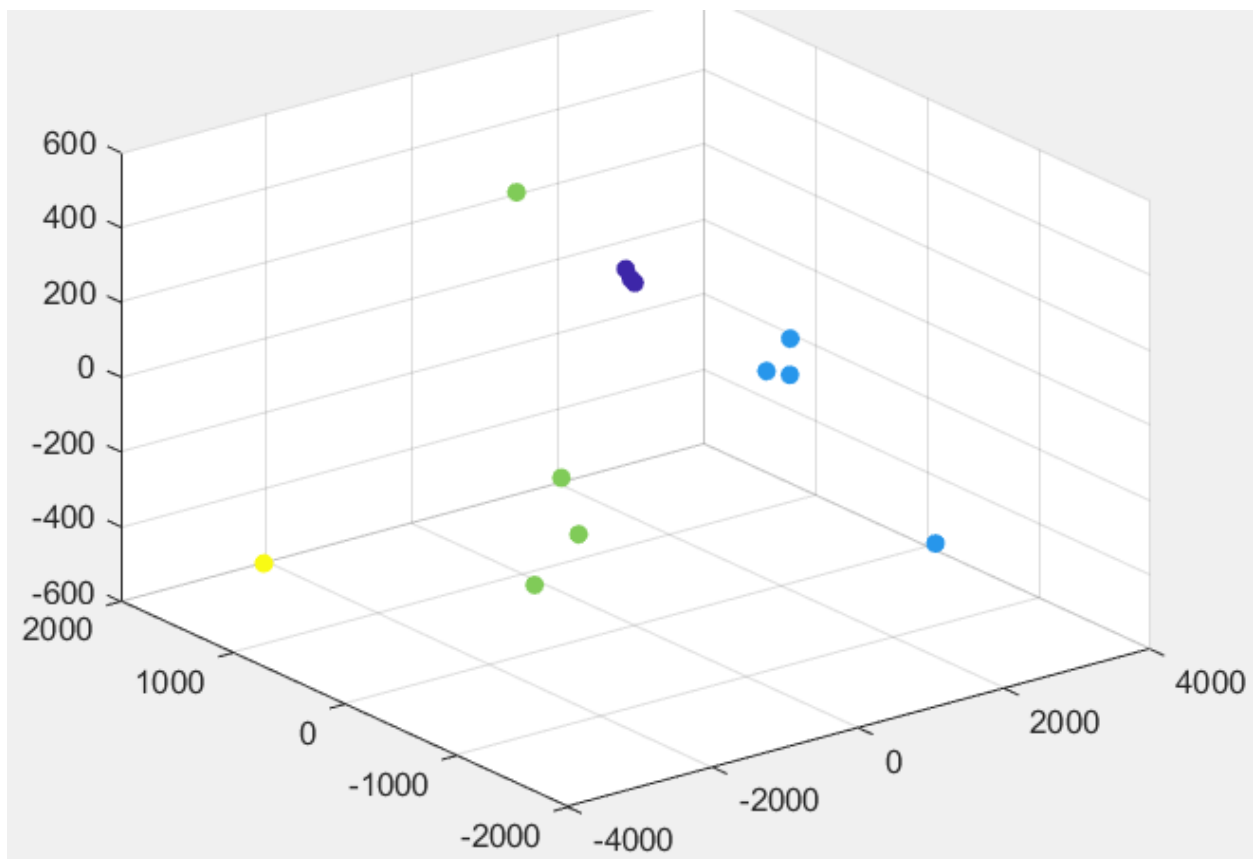
Experimental Results

Principal component analysis is applied to the test matrix of size 12x15. The ‘explained’ output of pca matlab command is the percentage of the total variance explained by each principal component. We can see that the first three components contribute to a variance of more than 95%, thus the first three components are chosen.

The percentage of total variance explained by each principal component

```
88.0372
9.4453
2.1341
0.2849
0.0843
0.0115
0.0024
0.0003
0.0000
0.0000
0.0000
```

The `pca` command returns the principal components in score.



Plot of reduced 3-D feature vector in the feature space

Command Window

New to MATLAB? See resources for [Getting Started](#).

Cluster 1: Following Images

1
6
12

Cluster 2: Following Images

3
5
8
9

Cluster 3: Following Images

2
4
7
10

Cluster 4: Following Images

11

We can see that the image 2 and 3 are misclassified after PCA is applied, thus the **Error rate**=(Misclassified/Total images)= $2/12*100=16\%$. Thus, the classification accuracy is 84%.

The clusters that are obtained by passing 25D feature to kmeans algorithm.

Cluster 1: Following Images

1
5
6
8
9
12

Cluster 2: Following Images

3

Cluster 3: Following Images

4

Cluster 4: Following Images

2
7
10
11

We can see that the 6 image are misclassified without PCA, thus the **Error rate=(Misclassified/Total images)=6/12*100= 50%. Thus, the classification accuracy is 50%.**

Thus, feature dimension reduction decreases the complexity of the classifier and the computational time. Thus, it preserves the most relevant information of the original input data. PCA performs a linear transformation moving the original set of features to a new space composed by principal component. PCA chooses those properties that shows as much variance across the classes to build the principal component. The eigen vectors represent the new axes and the eigen values gives the variance. Thus, in order to reduce the dimension, the eigen vectors with the more variance are chosen. **We can conclude that when the features are reduced from 15D to 3D, the classification accuracy improved.**

(b)Supervised

Procedure:

- The input training images, and testing images are read.
- The energy feature vectors are produced using the previous method.
- The PCA is applied to both the training energy vector and testing energy vector.
- In random forest classifier, the Treebagger is used. TreeBagger grows the decision trees in ensemble and reduces overfitting and improves generalization
- Treebagger returns an ensemble of NumTrees based on the inputs the pca reduced data and the train labels.
- The model is used to predict the test labels for the pca reduced test data.
- **The classification accuracy is 91.667 percent for RF**

In the case of SVM

- Train the ECOC classifier using SVM binary learners and standardize the predictors.
- **The classification accuracy is 58.33 percent for SVM**

Command Window

New to MATLAB? See resources for [Getting Started](#).

```
>> supervised
The percentage of total variance explained by each principal component
    86.1175
     9.2778
     3.5124
     0.9381
     0.0981
     0.0468
     0.0046
     0.0028
     0.0016
     0.0003
     0.0000
     0.0000
     0.0000
     0.0000
     0.0000

Random Forest Classifier Accuracy:
    91.6667

SVM Classifier Accuracy:|
    58.3333

SVM Classifier Gaussian Accuracy:
    58.3333
```

Create a template for SVM binary classifiers, and specify to use a Gaussian kernel function

```
t = templateSVM('Standardize',true,'KernelFunction','gaussian');
SVMModel = fitcecoc(Tr,train_label,'Learners',t,'FitPosterior',true);
[label_testSVM,score] = predict(SVMModel,score_test(:,1:3));
accuracy_SVM=sum(label_testSVM==test_label)/length(test_label);

disp("SVM Classifier Accuracy:")
disp(accuracy_SVM*100)|
```

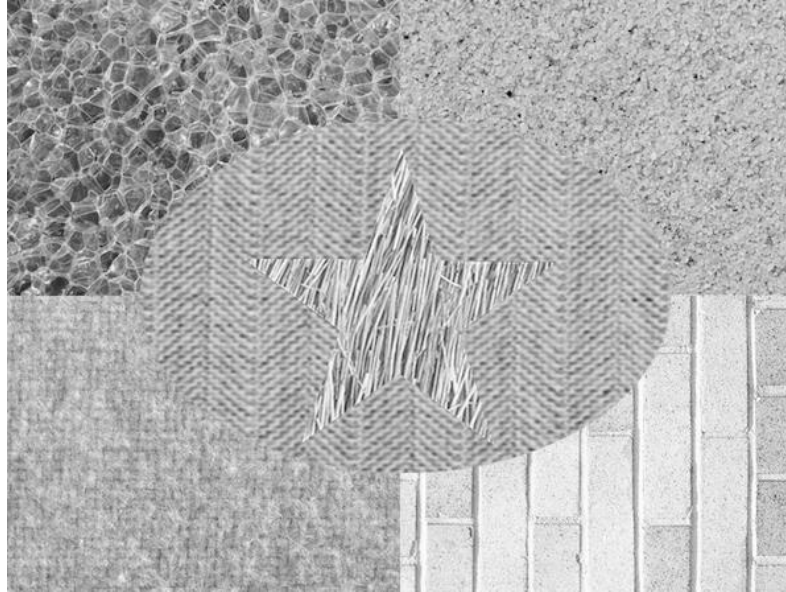
Discussion

We can see that Random Forest Classifier produces a better result compared to SVM .

(c) Texture Segmentation

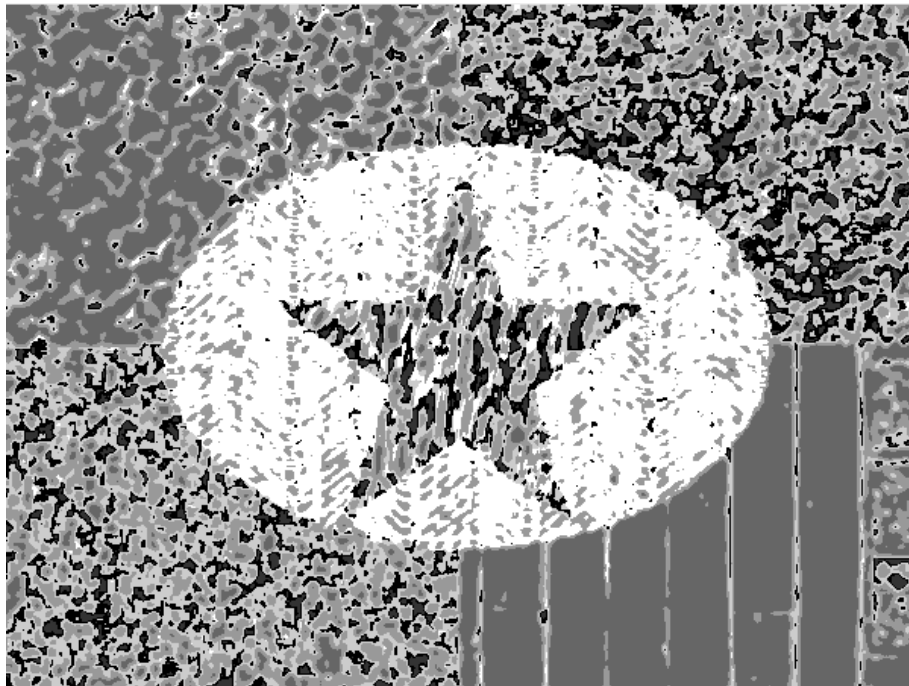
Procedure

- The input image Composite.raw image is read.

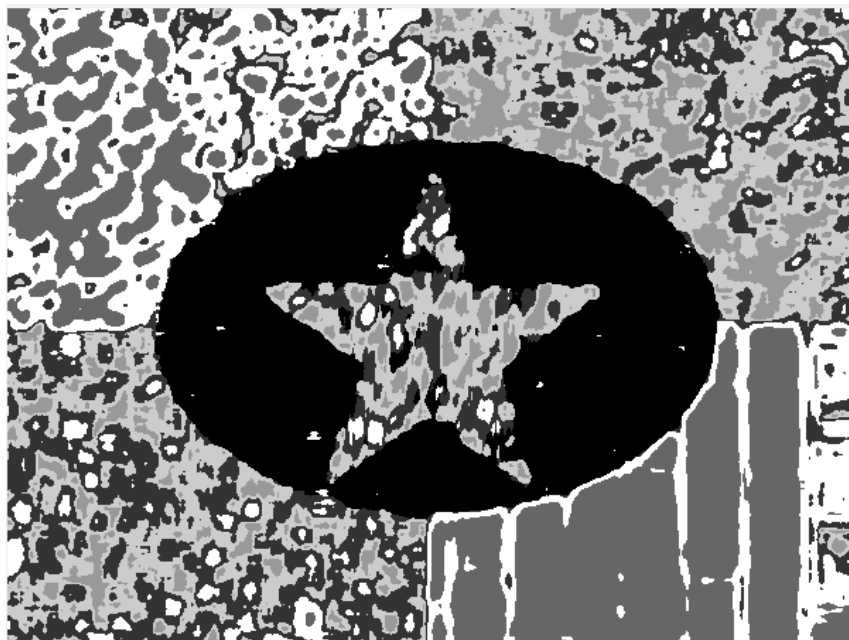


- In order to reduce the illumination effects, the image is subtracted by the mean of the entire image.
- The 2D 5x5 law filters are computed by the tensor product between each 1D kernels.
- The image is filtered using the law filters. Thus, producing different textures.
- The energy feature vector is computed which is equal to the square root of the absolute sum of the filter output value at each pixel. Thus, one image produces a (width*height)x25 vector. It produces a 25D energy vector.
- The 25D energy feature vector is reduced to 15-D energy feature vector by computing the average of every pair. (For example, the L5E5 and E5L5 have similar values).
- All the kernels have zero mean except L5E5. Thus, we can normalize the other feature by L5E5 to produce a 14-D energy feature vector.
- Since the image is 640x450 the final matrix will be (27000,15).
- The kmeans clustering is applied to the image produced in the previous step. Six different clusters are produced and are displayed using (0, 51, 102, 153, 204, 255) grayscale values.

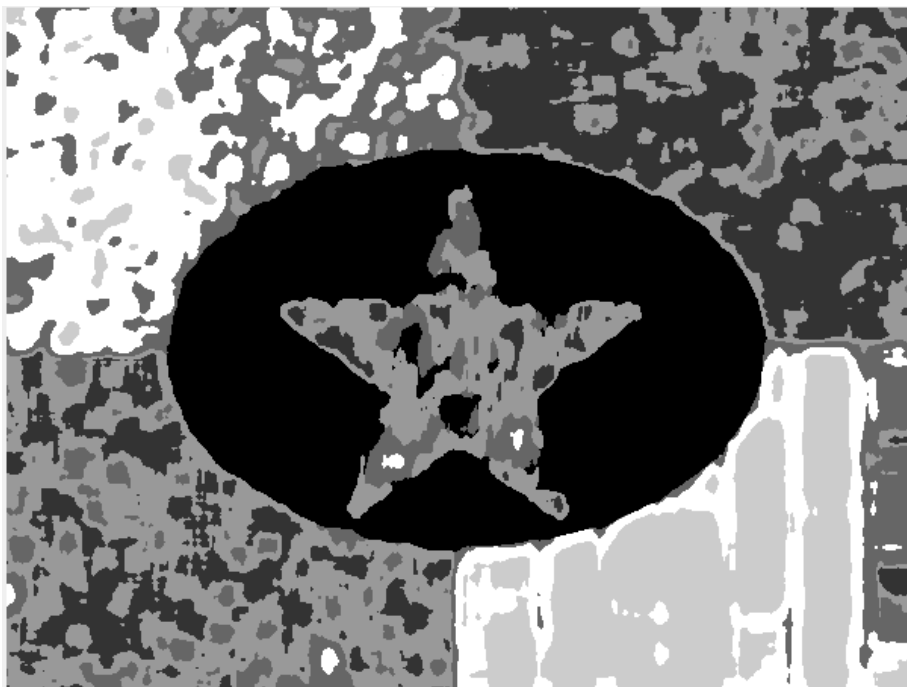
Experimental Results:



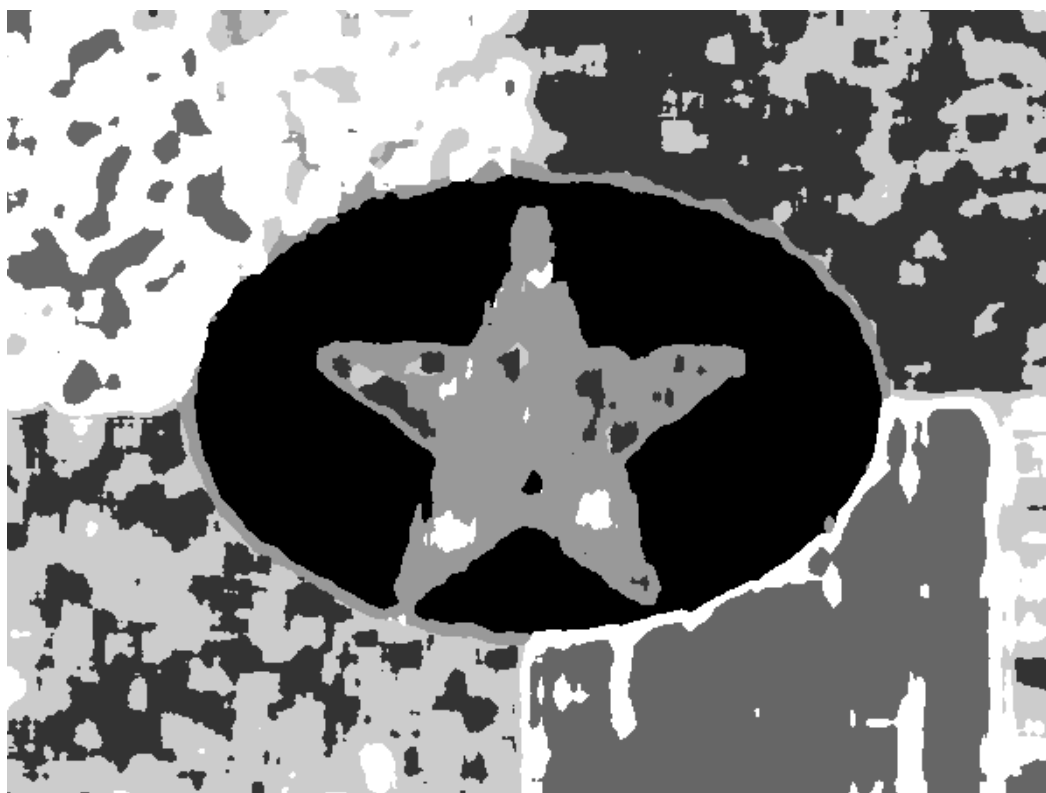
Window 5x5-14D energy vector



Window 11x11-14D



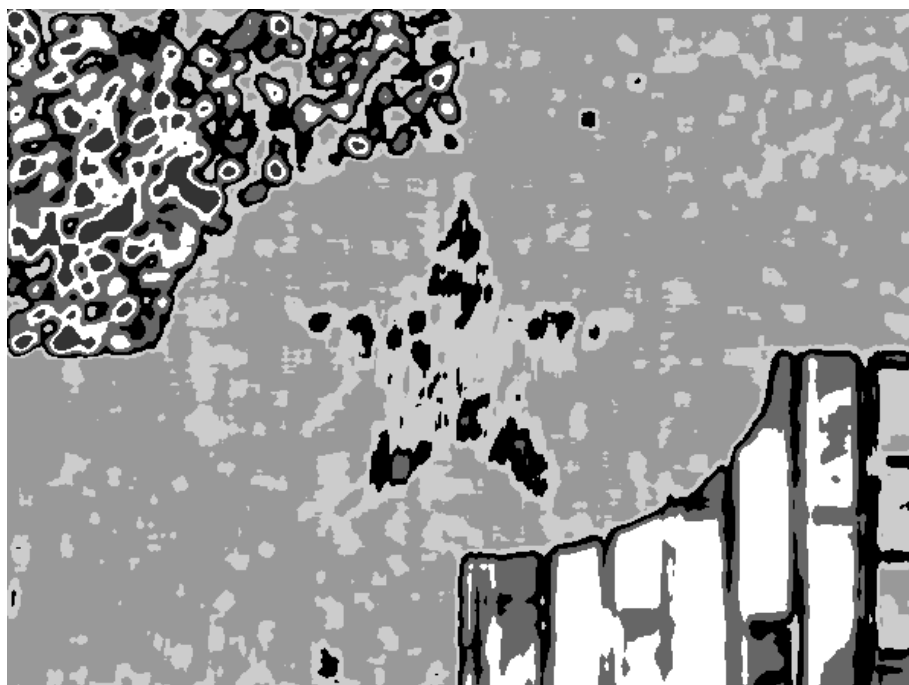
Window size: 13x13 to find the local energy feature with normalization



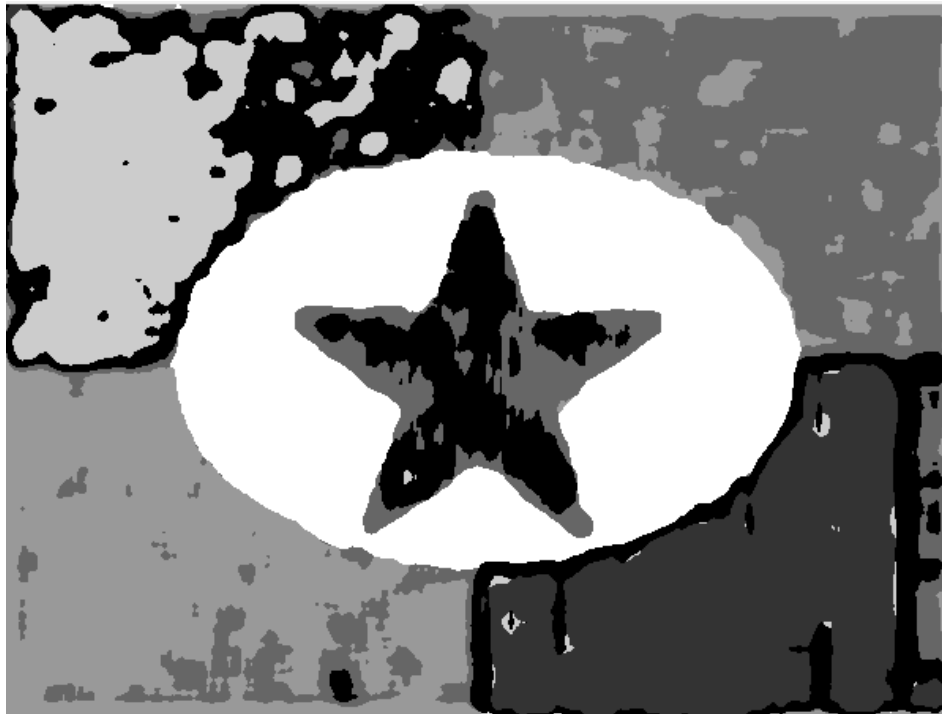
Window size 17x17 -14D



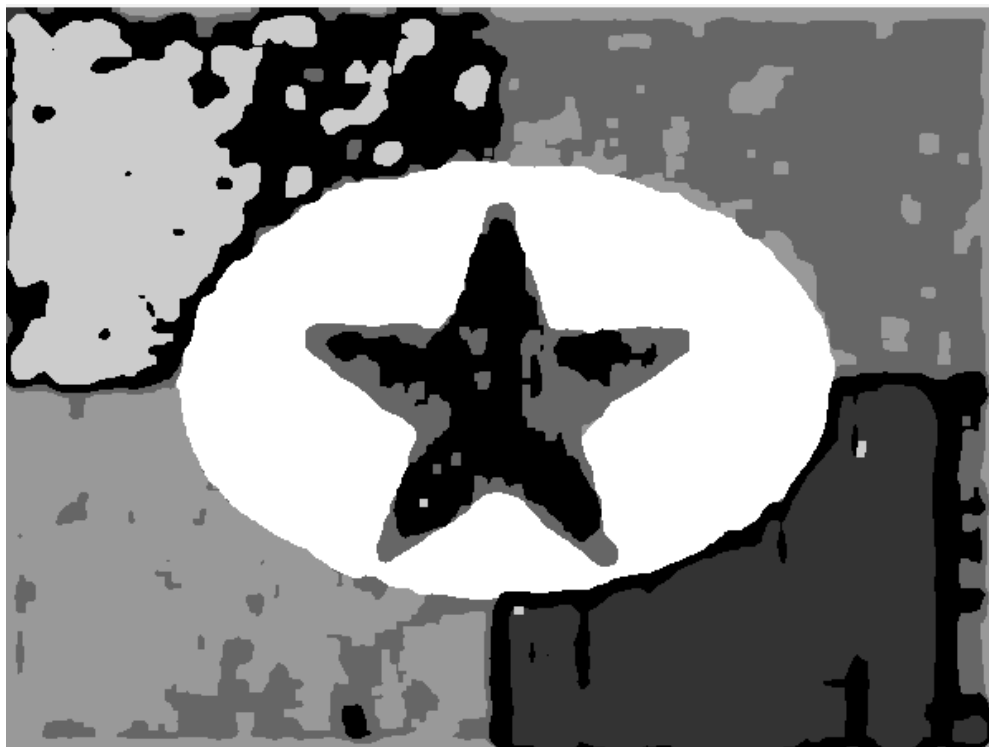
Window 19x19-14D energy feature vector



Window 13x13 without normalization



Window 19x19 without normalization



Window 19x19 Strel=3 without normalization



Window 19x19 without normalization



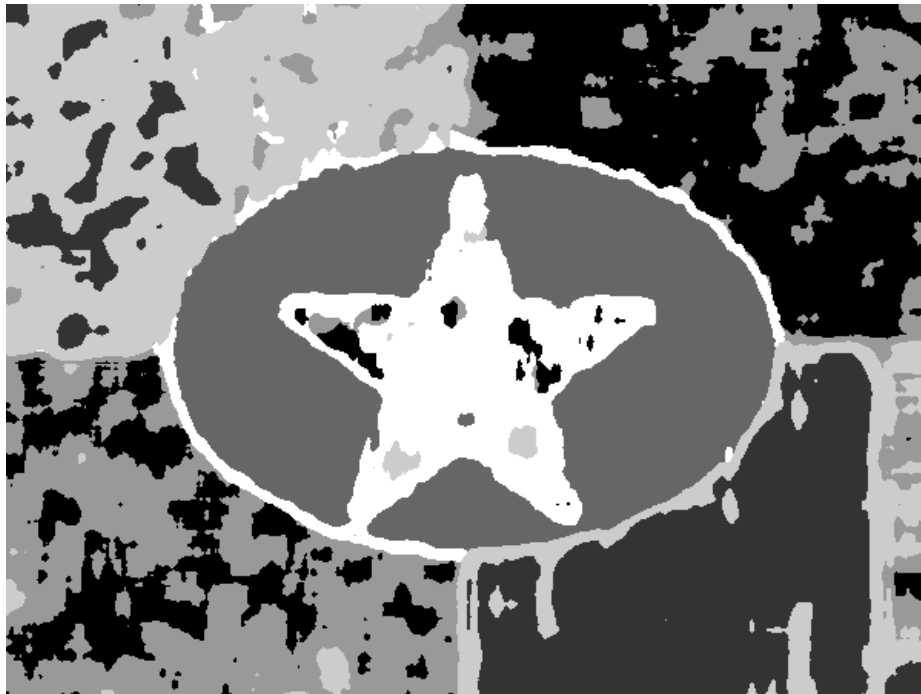
Window 19x19 without normalization with strel 3

Experimental Results

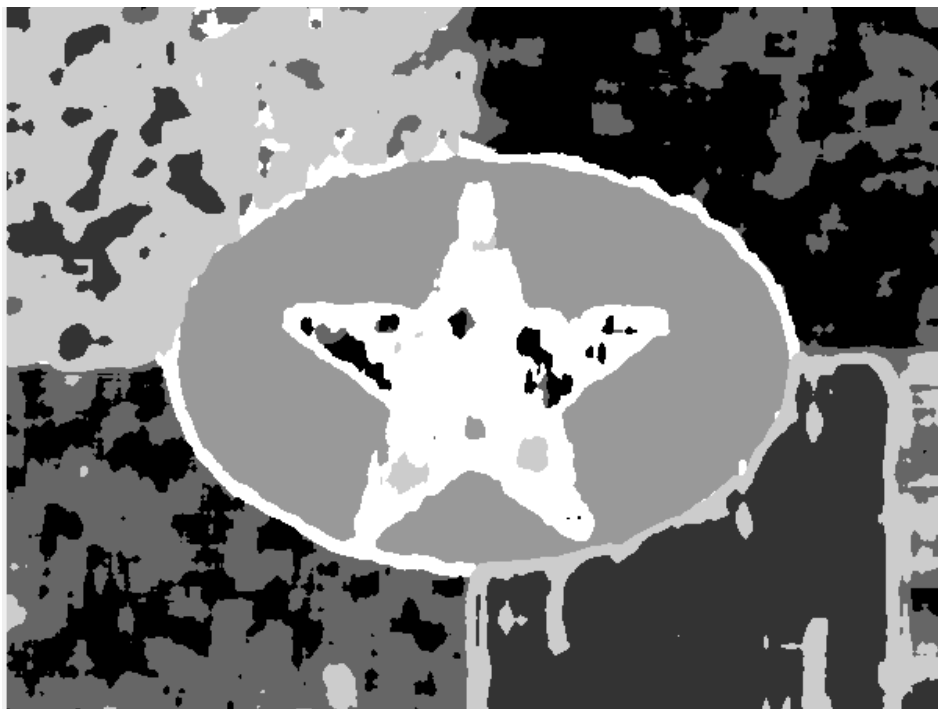
We can see that as the size of the window to compute the energy vector keeps on increasing the regions are more defined and the neighbouring pixels belonging to the same cluster are well defined. The minute areas are eliminated because the energy is computed across a bigger window. The normalized output gives a better result most of the times compared to unnormalized outputs. The best segmentation is produced for a window size 19x19. We can observe that unnormalized 14D energy features produce better results only when the window size is very big like 19x19.

(d)Advanced Texture Segmentation

PCA

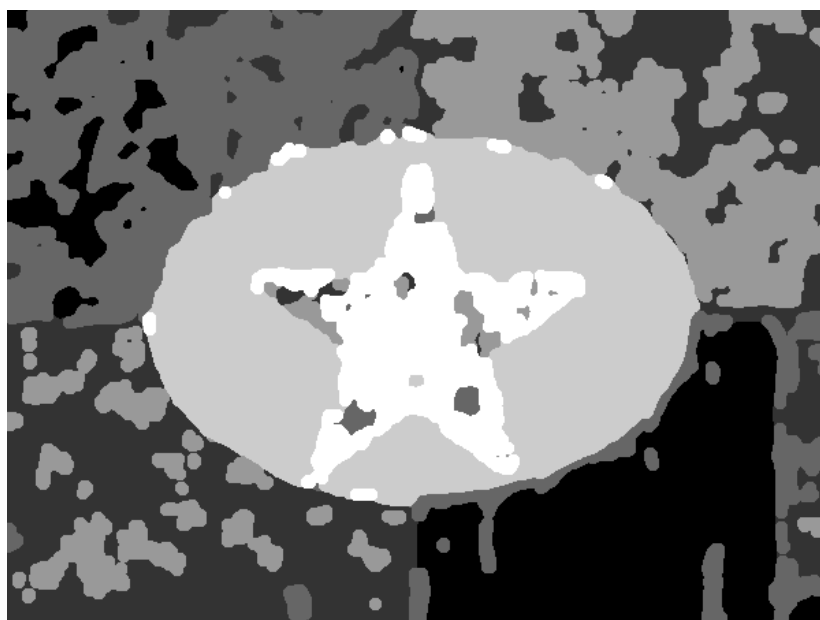


Window 19x19 PCA with 3 components



Window 19x19 with PCA 2 components

Morphological Operations

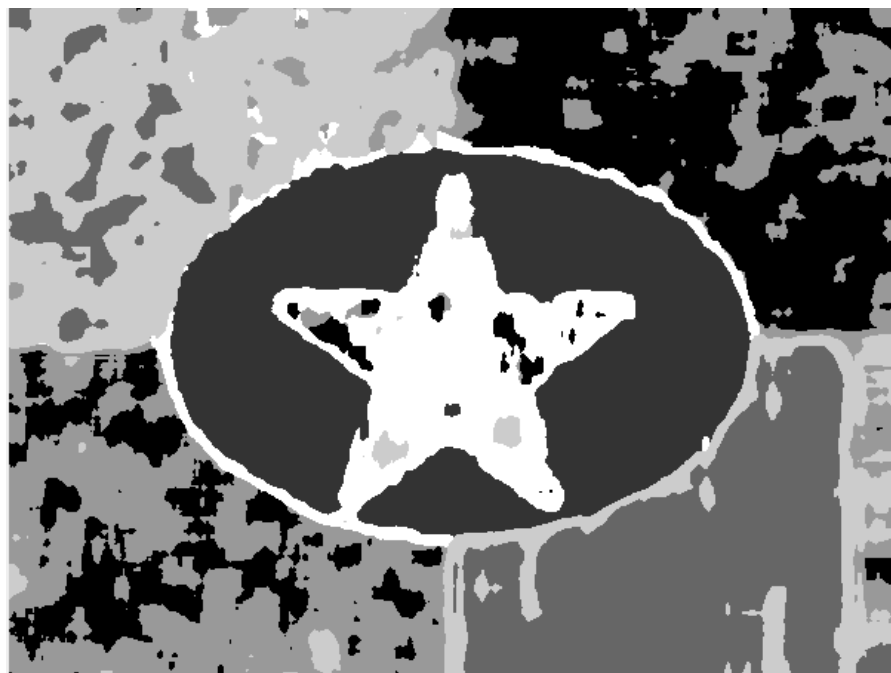


Strel=5 Morphological Opening

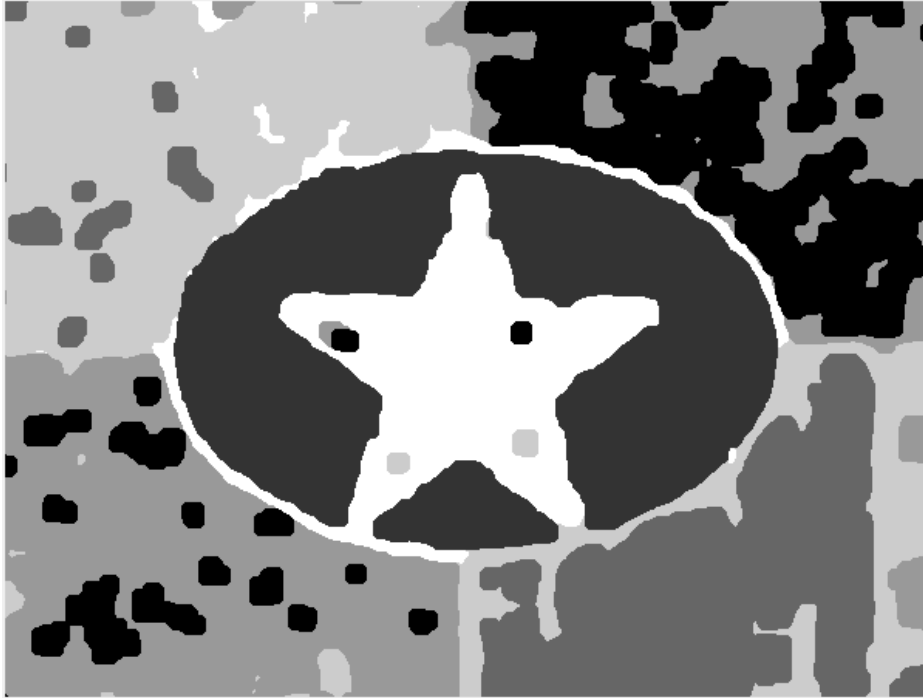


Strel=7 Morphological Opening

Best Output



Widow 19x19 with normalization



Widow 19x19 with normalization Closing with strel 'disk' =7



Morphological Opening with strel 'disk' =5 to the previous result

Experimental Results

We can see that as the size of the window to compute the energy vector keeps on increasing the regions are more defined and the neighbouring pixels belonging to the same cluster are well defined. The minute areas are eliminated because the energy is computed across a bigger window. In order to improvise the results, PCA was computed. Dimension reduction to 3D from 14D did not improvise the result much. Morphological operations such as opening the image is used. We can see that the result is better. With the size of the structuring element $strel=5$. Beyond that unwanted regions are combined. The best output is produced with Window size 19×19 and the image is closed using $strel 7$ and opened using $strel 5$. Thus, the minute holes the segmented region is eliminated.

Problem 2: Image Feature Extractors

(a) Salient Point Descriptor

1. The SIFT feature is invariant to scaling, translation and rotation and robust to affine transformations, change in 3D viewpoint, addition of noise and changes in illumination.
2. The DOG is difference of gaussian and its extrema depends on the value of $(k-1)$ and sigma square. Since, the extrema chooses the maxima or minima, the location does not depend on the scale. Thus, DOG is scale invariant. The image is rotation invariant because one or more orientations are assigned to each keypoints based on image gradients. Therefore, all the transformations on the image are relative to these orientations.
3. The descriptors are formed from 2×2 array of oriented histogram and each histogram has 8 bins. Thus, the 128-element feature vector is formed for each key point. The vector is normalized to unit length. When there is a change in the image contrast, each pixel value is multiplied by a constant therefore the gradients are multiplied by the same constant. This contrast change will be cancelled by the vector normalization. When there is a change in the brightness, each pixel value is added to a constant. Since the gradient is computed from difference in the pixels, this change would not affect the descriptor. Thus, the descriptor is robust to changes in illumination.
4. SIFT uses Difference of Gaussian (DOG) instead of Laplacian of Gaussian because DOG is scale invariant but LOG is not.
5. The output vector size is 128×40000 . 128 element feature vector and 40000 keypoints.

(b) Image Matching

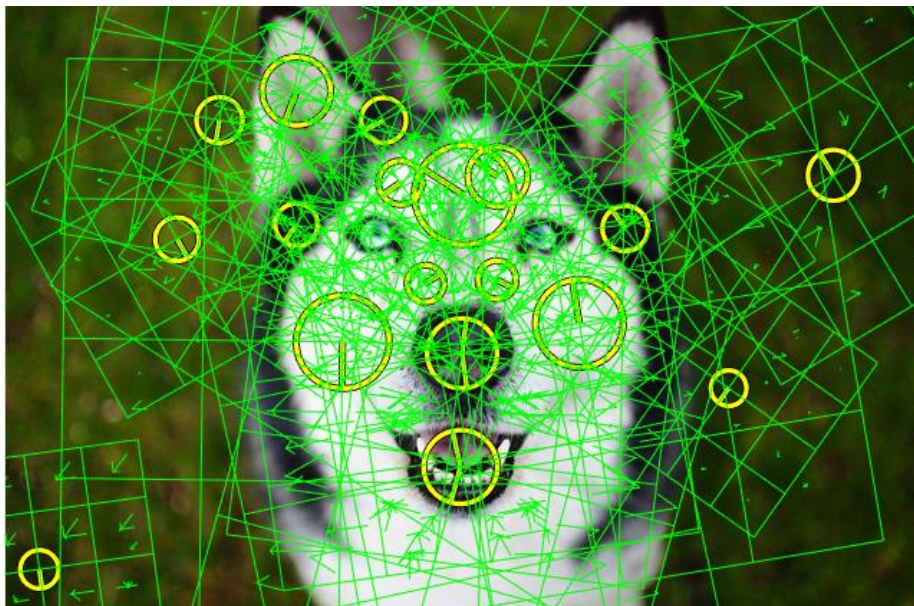
Procedure

- The input images are read.

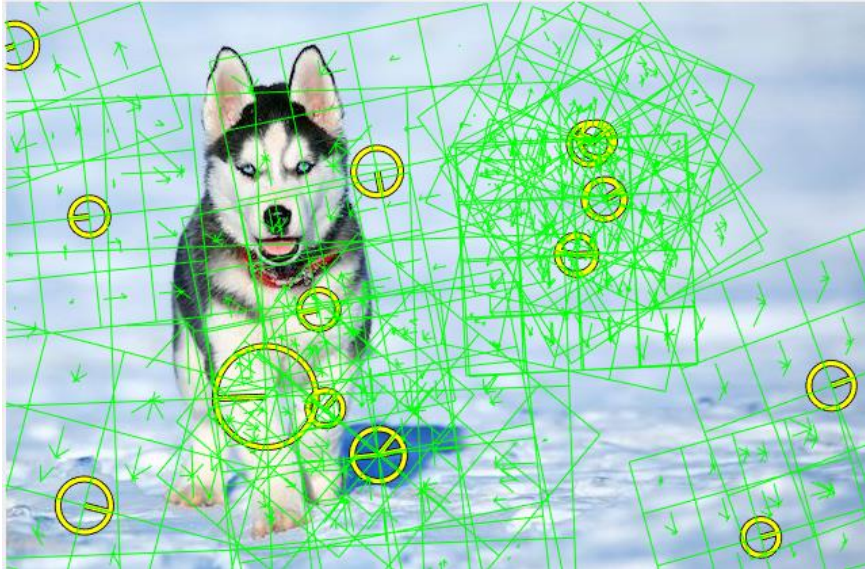
- The sift features are extracted. The sift features are mapped using `vl_ubcmatch` with different thresholds. The closest neighbor is chosen
- The matched features are mapped using RANSAC.
- The matched features are plotted.



Descriptor with the max scale



Descriptors in Husky 3



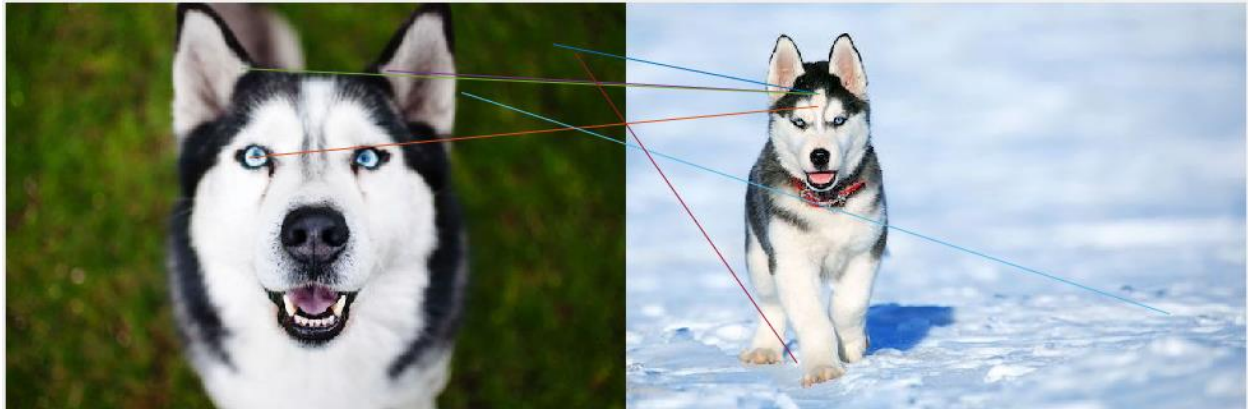
Descriptors in Husky 1

The keypoint frame gives the descriptors center, its size and the orientation.

Husky 3 and Husky 1

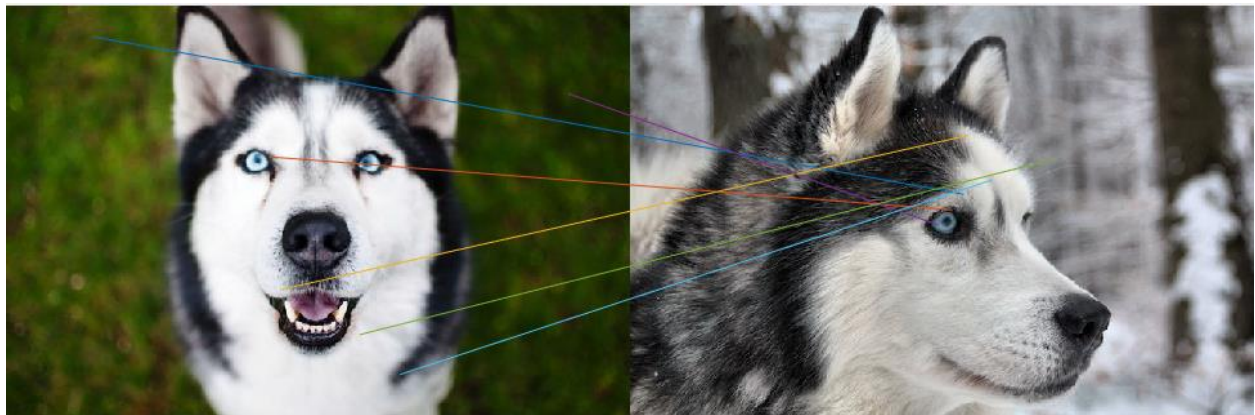


v1_ubcmatch with threshold 2



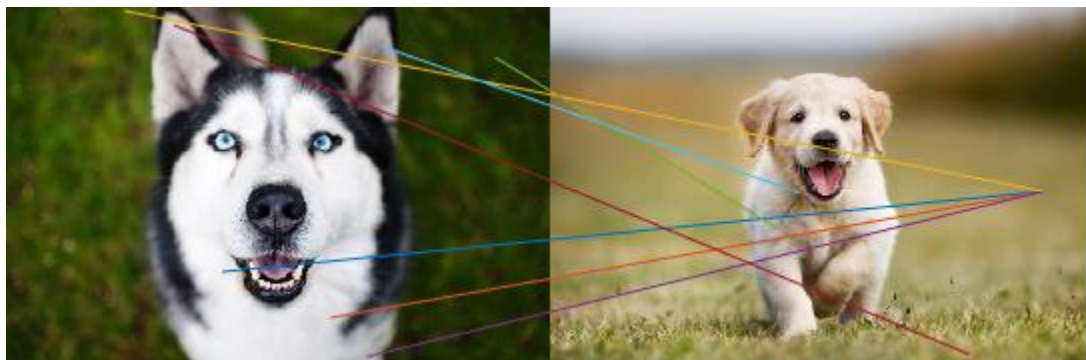
vl_ubcmatch with threshold 1.2

Husky 3 and Husky 2



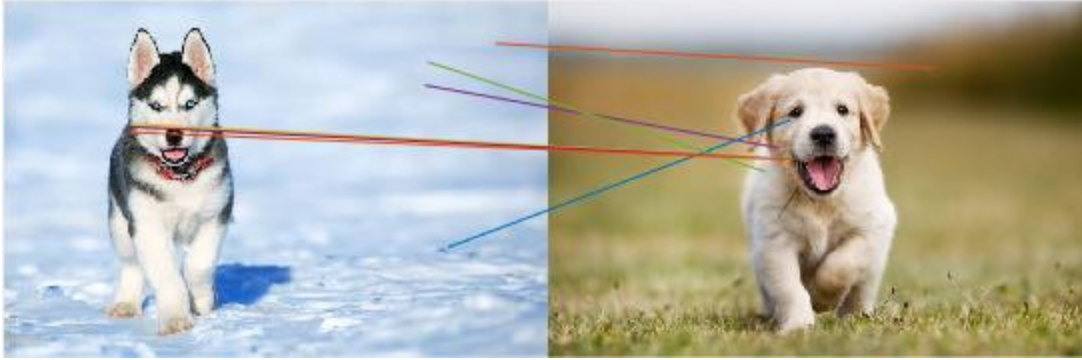
vl_ubcmatch with threshold 1.2

Husky 3 and puppy 1



vl_ubcmatch with threshold 2

Husky 1 and puppy 1



We can see that the image that matches the best is Husky 1 followed by Husky 2 and Puppy 1 with reference to Husky 3. Husky1 with Puppy 1 gives a better match because the husky image gives more resemblance to a walking dog with the background unlike Husky 2.

(c)Bag of Words

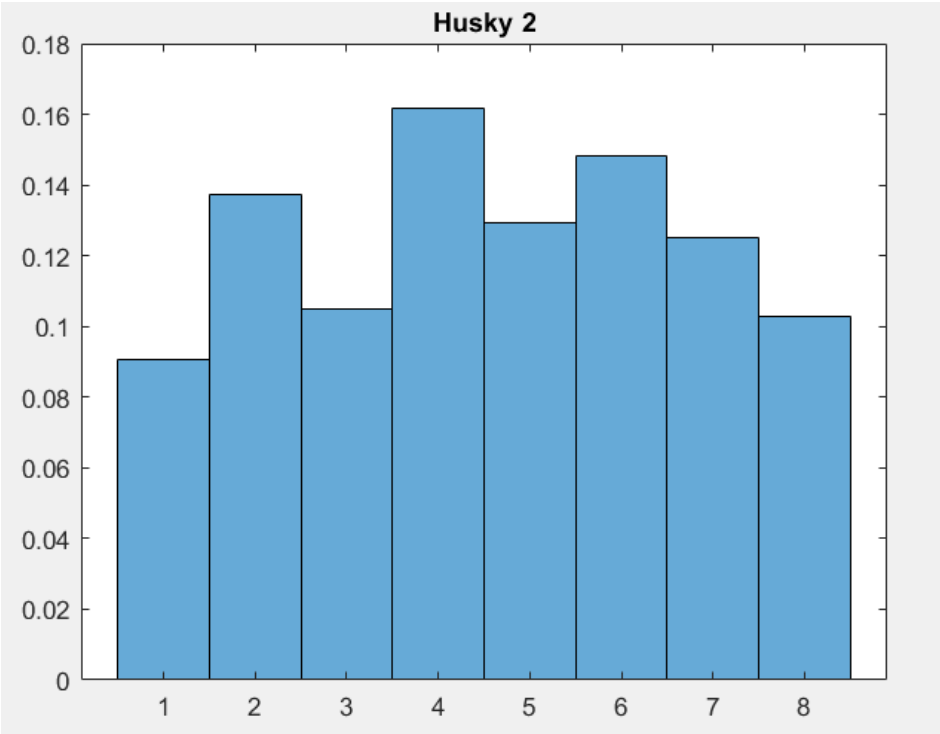
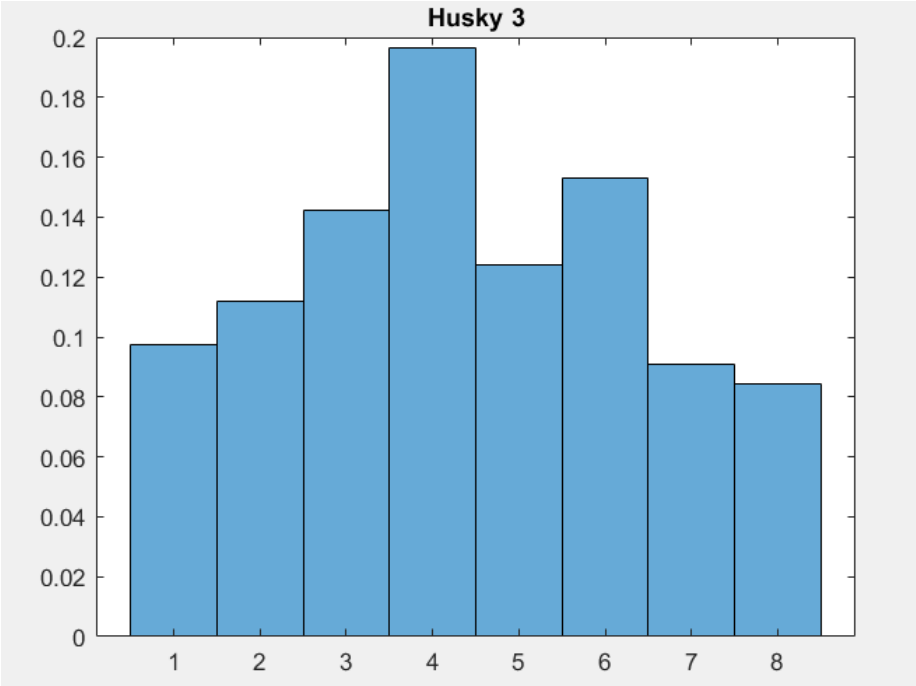
Bag of Words are used to match images using the concept of histogram of codewords.

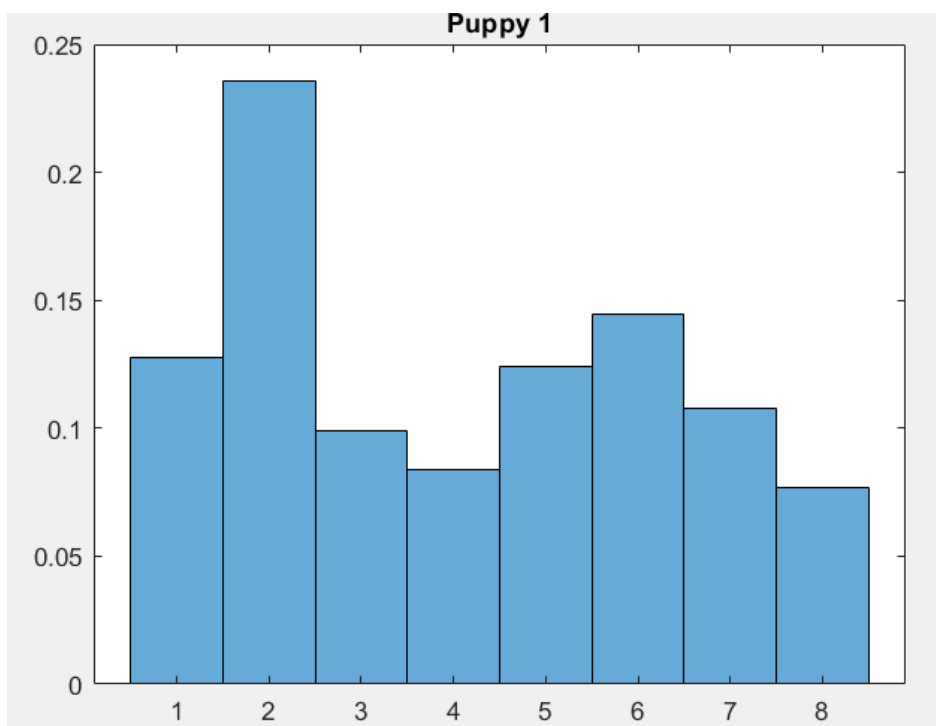
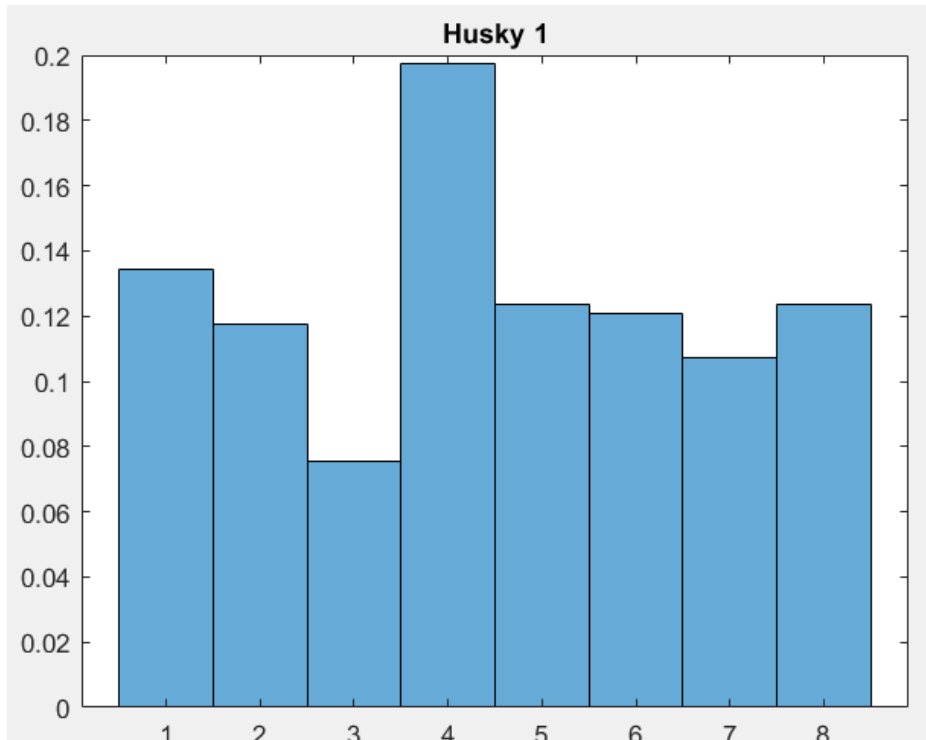
Procedure:

- Read the four images. The image Husky 3 is considered as a reference image.
- The SIFT features are extracted from each image using vl_shift.
- The vl_shift gives the frame(keypoints) and the descriptors.
- Kmeans algorithm is applied in order to group the descriptors into 8 bins.


<https://www.vlfeat.org/overview/kmeans.html> [Source]

- Thus, the histogram of Husky 3 is computed with 8 centroids and the count of the descriptors.
- Similarly, the histogram of Husky 1, Husky 2 and Puppy 1 are created.
- Histogram matching is done by computing the absolute difference between the sorted values of the histogram. The histogram that produces minimum error is considered to be similar to the Husky 3 histogram.





The reference image is Husky 3. From the histogram we can see that husky1 is the best match compared to the other images. The second best is Husky2 and the last best is Puppy1.

	diff1	0.0373
	diff2	0.0364
	diff3	0.0728

We can see that the error is minimum for diff2(Husky 1) followed by Husky2(diff1) and Puppy1(diff3).