EE569 Homework 4

Problem 1: Texture Analysis

1(a) Texture Classification

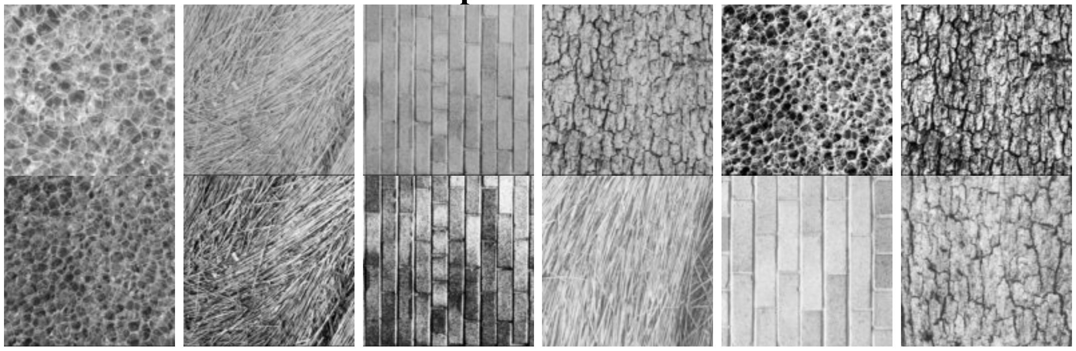
Motivation:
We are doing unsupervised classification of textures using Law's filters. Even though we have advanced methods for texture classification, Law's filters give us a basic understanding using different filters.

Approach:

1. Create Law's filter bank of 25 filters using tensor product of following kernels:
    - Level [1,4,6,4,1],
    - Edge [1,-2,0,2,1],
    - Spot [-1,0,2,0,-1],
    - Wave [-1,2,0,-2,1],
    - Ripple [1,-4,6,-4,1]
2. Read the image and subtract the mean of all pixels from image so as brightness of image does not affect the classification.
3. Extend boundary of the image by 2 as we are applying filter of window size 5.
4. Find 25 filtered images using 25 filters as masks to this extended image.
5. Find Energy of each filtered image using the average of absolute values of all pixels in the image.
6. Now, you should have 25 Energies for one image. This acts like feature vector for the image.
7. Standardize these energies by subtracting by mean and dividing by standard deviation.
8. Repeat from step 2 for all 12 images and store these energies in a matrix.
9. Use Principal Component Analysis to reduce the 25-D vector to 3-D vector.
10. Use k-means clustering to cluster these 12 images into 4 clusters.

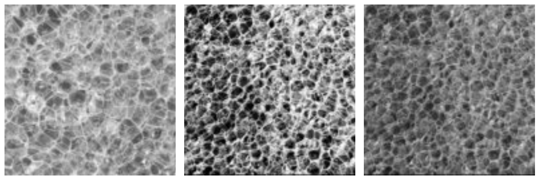
Experimental Results:

Input textures:

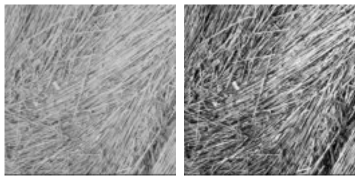


Output:

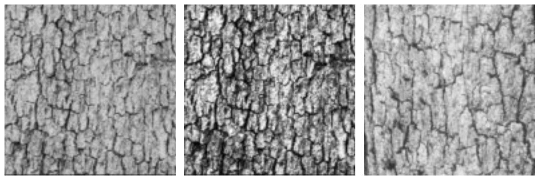
Class 1:



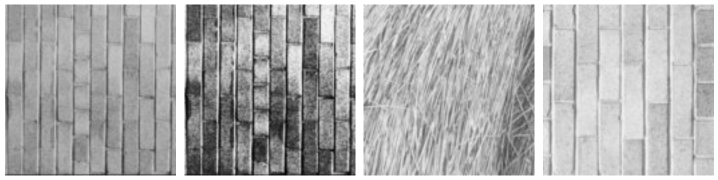
Class 2:

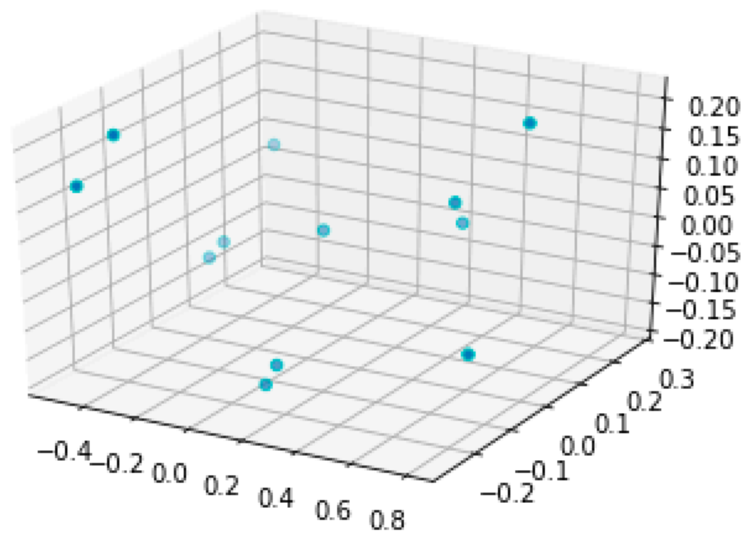


Class 3:



Class 4:





3D plot of feature vector in feature space

Discussion:

- We have one misclassification according to the above method.
- Strongest and weakest discriminant power:
  Feature (L5)TxL5 has weakest discriminant power. (S5)TxS5 has strongest discriminant power. Discriminant power of feature with high variance is low in the direction of feature vector. And, vice versa.
- We get same classification with and without using PCA.
  We have 25D feature vector. Which is high dimensional data. So, we project it into lower dimension and check percentage of variance described by each component. After dimensionality reduction, the first 3 components have following explained_variance_ratio:
  0.804598, 0.112689, 0.0619722 which amounts to a total 97.92 of the total variance. The rest are close to zero. Hence, we can work with a 3D vector instead of 25D vector and get similar results. It is easy for visualization as well. And Classification works better for reduced dimension dense data than sparse multidimensional data.

1(b) Texture Segmentation

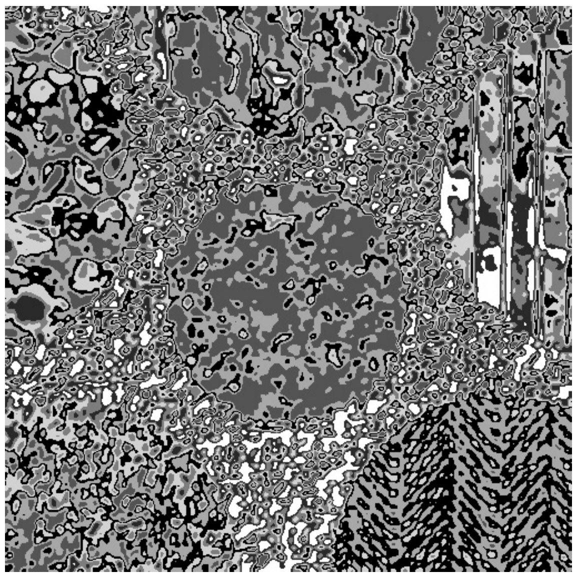
Motivation:
Sometimes we have images with different textures within the same image. We need to segment such image. It can be used to remove unnecessary features or details in background from an image.

Approach:

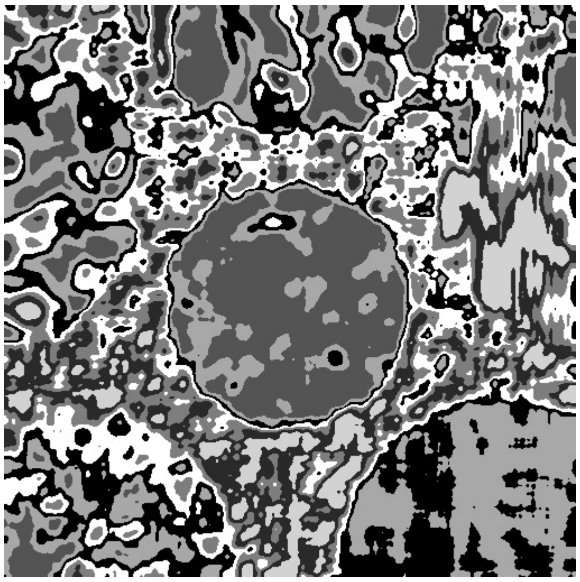
1. Create Law's filter bank of 25 filters using tensor product of following kernels:
    - Level [1,4,6,4,1],
    - Edge [1,-2,0,2,1],
    - Spot [-1,0,2,0,-1],
    - Wave [-1,2,0,-2,1],
    - Ripple [1,-4,6,-4,1]
2. Read the image and extend its boundary by 2 for 5x5 filters from the filter bank
3. Find 25 filtered images using 25 filters on each pixel of the image.
4. Pad the image by 8 for 15 size filter.
5. Find energy of each pixel using average of absolute values of pixels in a 15-size window for all 25 images.
6. Divide all the energies by the energy obtained by (L5T,L5) to normalize all other features.
7. Use k-means clustering with 7 clusters. Assign different pixel values to the 7 labels. Ex. 0 → 0 , 1 → 42, 2 → 84, 3 → 126, 4 → 168, 5 → 210, 6 → 255

Experimental Results:

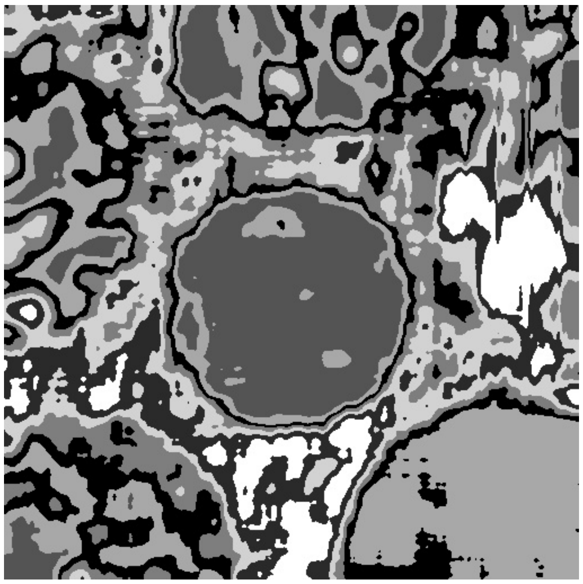
Window size 5:



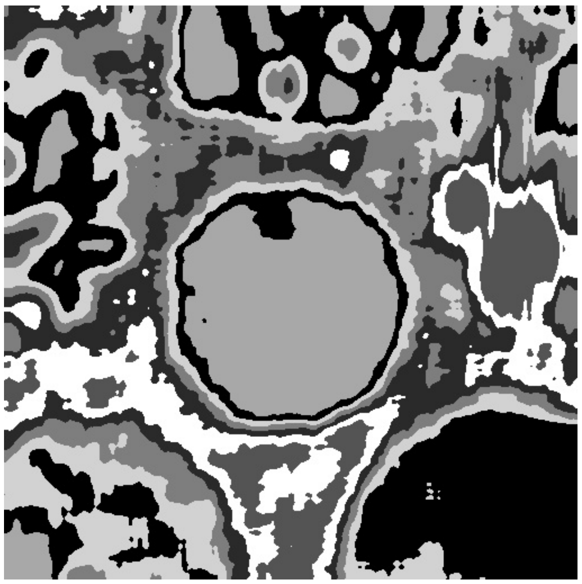
Window size 15:



Window size 25:



Window size 35:



Discussion:

- Window of size 35 gives best result.
- We have used (L5)TxL5 to normalize the filtered images through rest filters as the discriminant power of (L5)TxL5 is lowest as seen from 1a.
- Window size of 5 is very small to discriminate any textures. It gives wormy effect to the image, highlighting each small detail.
  As the window size increases, the image starts to get smoother and we can vaguely see different textures. I have used some techniques to modify these results further in 1c.

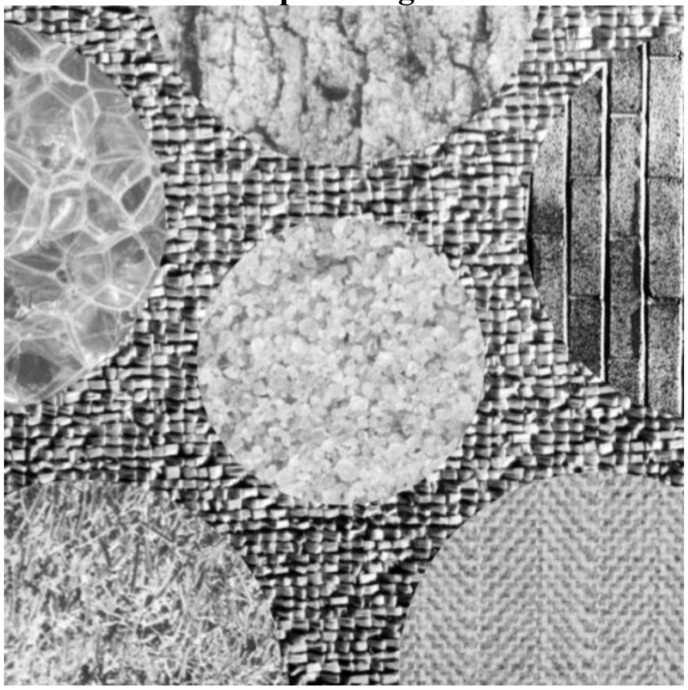
1(c) Advanced Texture Segmentation Techniques

Motivation:
We see that the images aren’t segmented properly. Hence, we try advanced techniques to make the segmentation more presentable.

Approach:
1. Use output of 1b
2. Standardize the energy features by subtracting the mean and dividing by standard deviation.
3. Use PCA to reduce the dimension to 3.
4. Use morphological closing to fill holes.

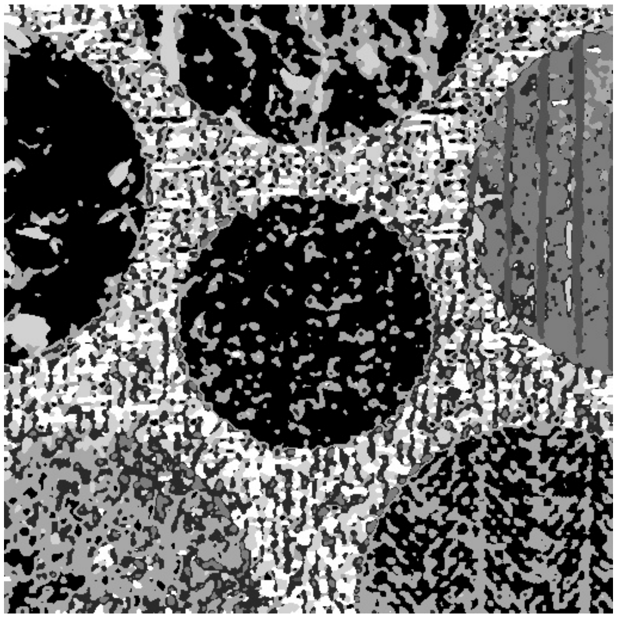
Experimental Results:

Input Image:

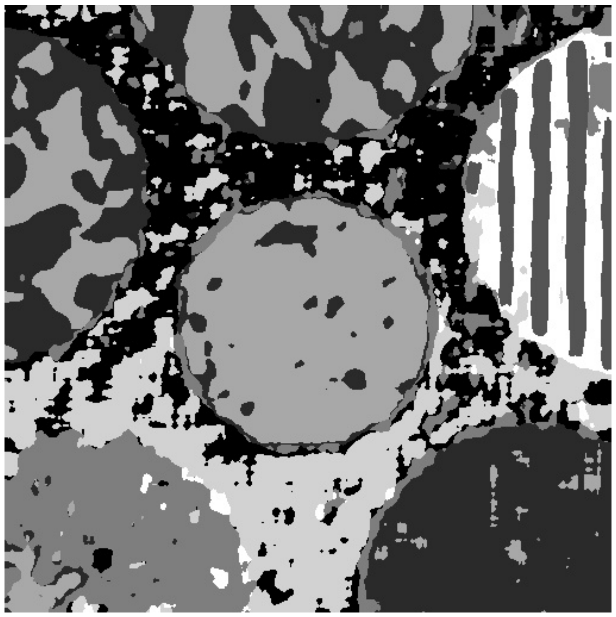


With Standardization and PCA:

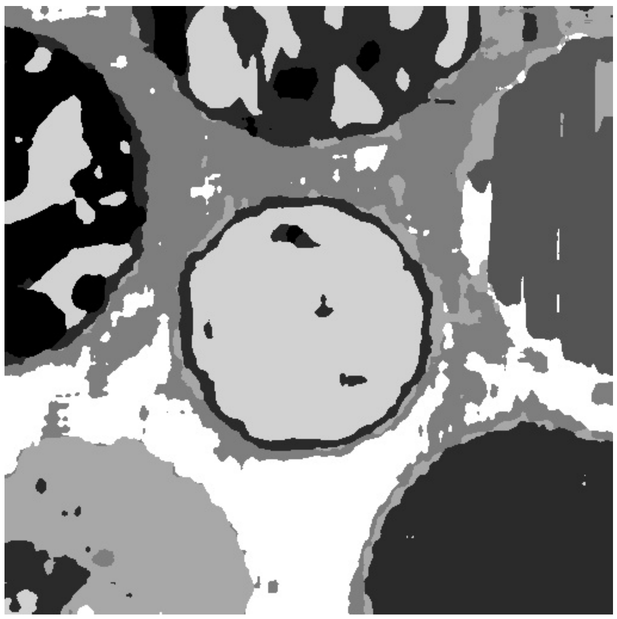
Window Size 5:



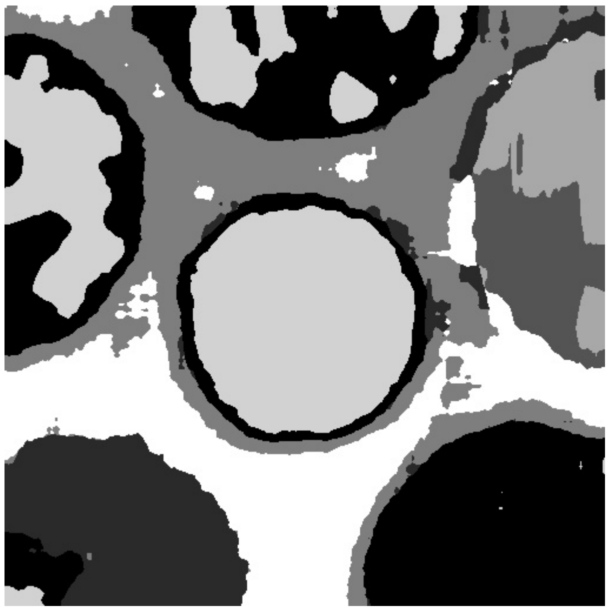
Window Size 15:



Window Size 25:

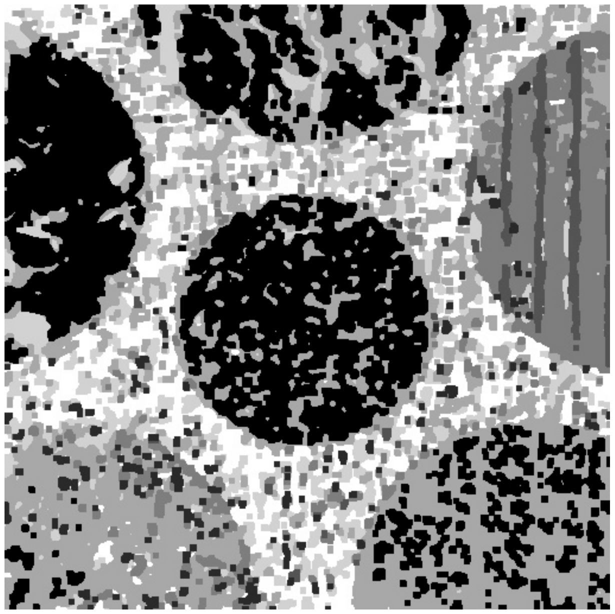


Window Size 35:

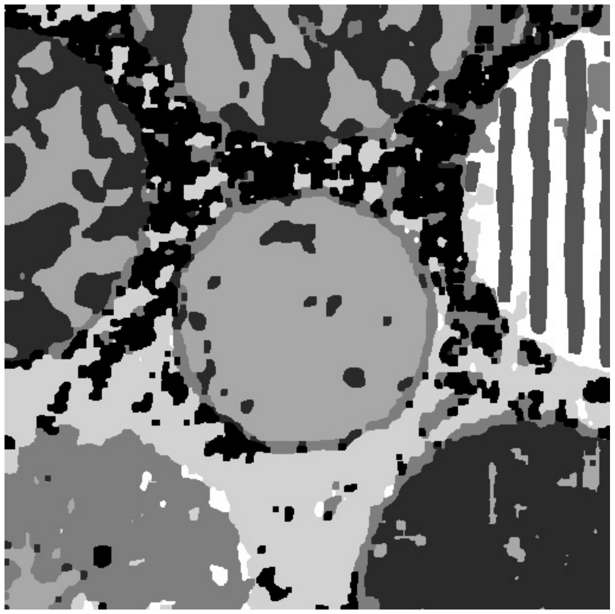


With Standardization, PCA and closing:

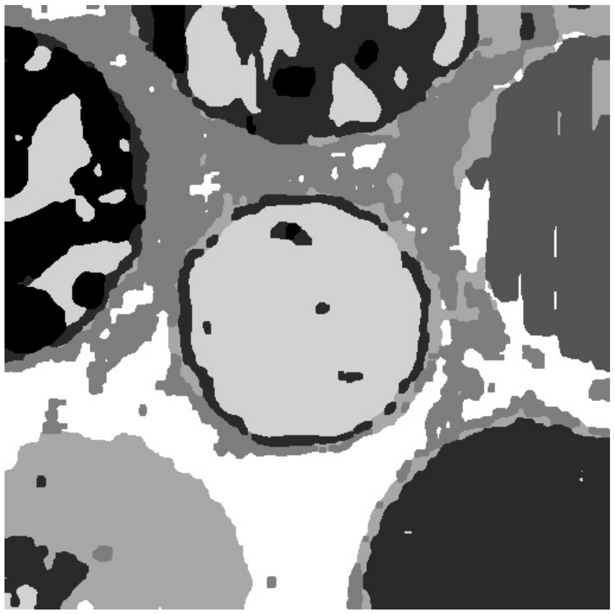
Window Size 5:



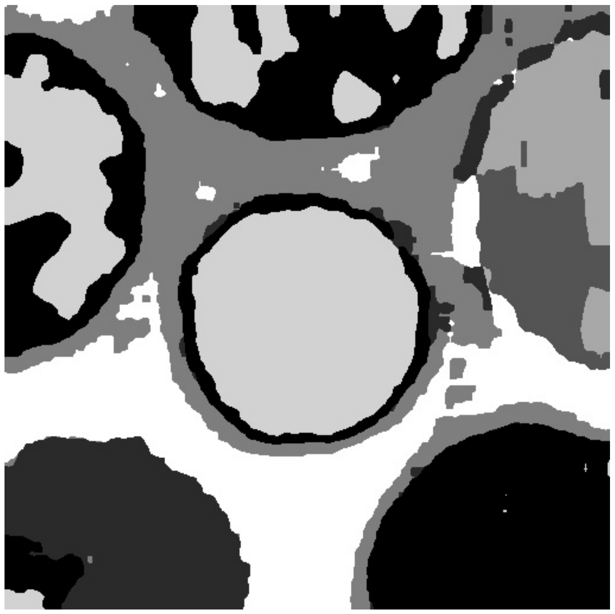
Window Size 15:



Window Size 25:



Window Size 35:



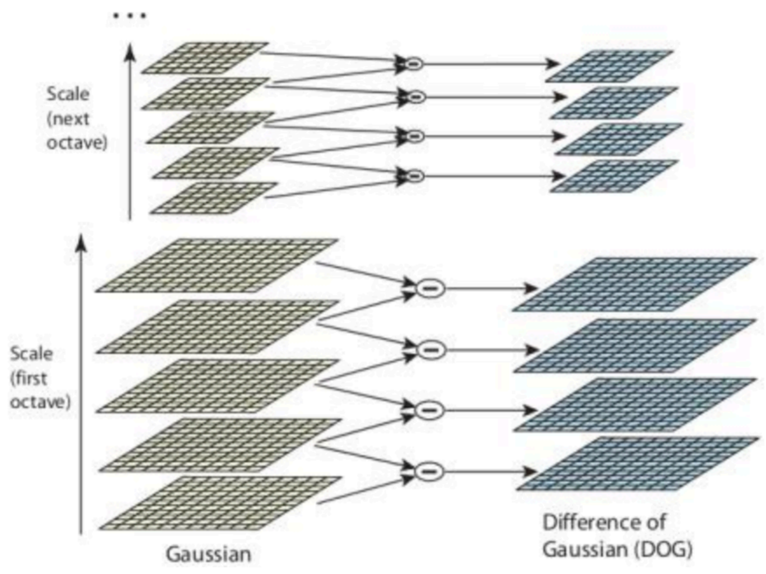
Discussion:

- PCA with dimension 3 has variance ratios 0.728366, 0.188479, 0.0534896 amounts to a total 97.03 of the total variance.
- Holes are removed slightly.
- We can see that textures look classified and have different colors and boundaries are also highlighted.

Problem 2: Image Feature Extractor

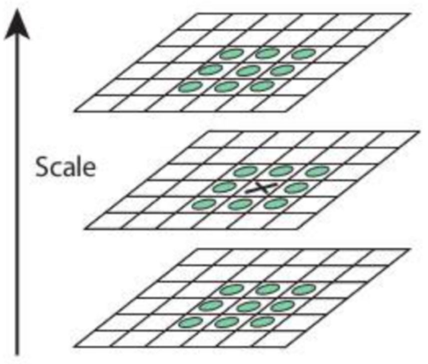
2(a) SIFT (From the paper)

1. SIFT is invariant to geometric modifications like image translation, image scaling, image rotation. It is also partially invariant to change in illumination, affine and 3D projection.
2. We use **Scale- space Extrema Detection** for **robustness in scale**:
   Laplacian of Gaussian of image with different sigma values is used for determining scaling factor for window for corner detector. Gaussian kernel with small sigma fits well for smaller corner and larger sigma gives a large value for large corners. We can find key point by looking for local maxima in (x,y, sigma) space.
   Paper uses Difference of Gaussian Kernels instead of Laplacian of Gaussian due to the high cost of LoG. We blur an image with 5 different sigma values and take difference pairwise. Now, we resize the image into half and repeat the process. We repeat this till a pyramid is formed.



image

Now we find the extreme for searching in both scale and space. As shown below, every pixel is compared to 8 surrounding neighbors in same scale and 9 surrounding pixels in the scale above and below. If the pixel is extrema within these, it is a potential key point.



image

For robustness in rotation,
Every pixel has image gradient M(I,j) and orientation R(i,j):

$$M_{ij} = \sqrt{(A_{ij} - A_{i+1,j})^2 + (A_{ij} - A_{i,j+1})^2}$$

$$R_{ij} = \operatorname{atan2}(A_{ij} - A_{i+1,j}, A_{i,j+1} - A_{ij})$$

where A(i,j) is the image pixel magnitude.
Orientations are divided into 36 bins which angles 10, 20, 30 .. 360 degrees. These orientations are weighted with their gradient magnitudes and added in bins. The gradients are themselves weighted with a gaussian distribution about the key point with sigma of 1.5* scale-sigma.

One key point can have multiple orientations. We keep all orientation above 80% of the highest orientation. This is how it assures robustness in rotation.

3. For robustness in illumination, SIFT calculates descriptor vector for each key point. For every key point, a 16x16 neighborhood is considered. It is divided into 4x4 sub regions. Each sub region has 8 bins each(8 directions in one plane). Thus total of (16*16)/(4*4) regions with 8 bins each make a total of 128 descriptors. This vector is further normalized to achieve robustness in illumination. For change in illumination in 3D, robustness is enhanced by thresholding gradient magnitudes by 0.1*maximum gradient magnitude.
4. LoG is quite costly computationally. Hence, we use DoG instead.
5. Original vector size was 160.

2(b) Image Matching

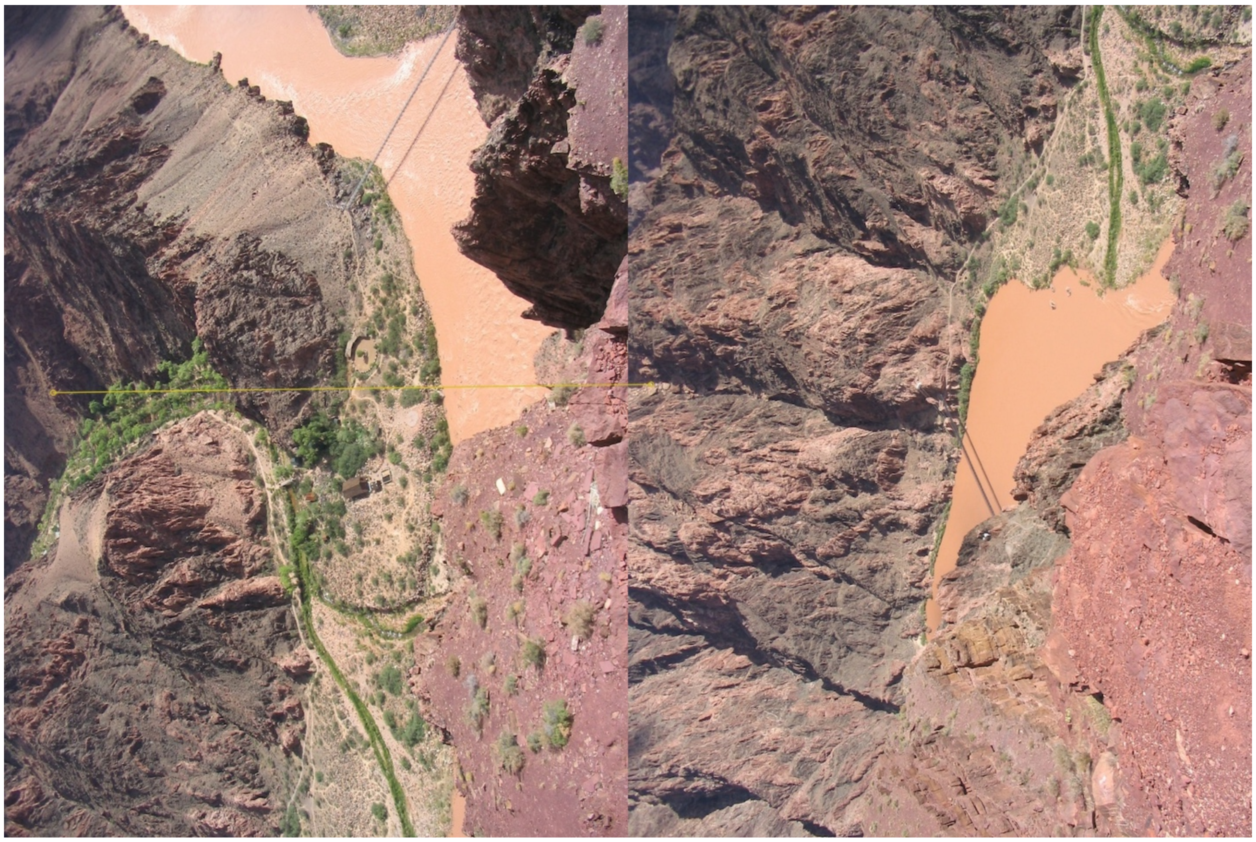
Motivation:

Using SIFT makes the image invariant to image translation, image scaling, image rotation. Thus, the obstacle of object detection due to change in size, shape is solved. It has various applications like object detection.

Approach:

1. Load both river images and find their key points and descriptors using SIFT.
2. Find the l2 norm of descriptors of each key point of first river image.
3. Find the key point corresponding to the largest l2 norm.
4. Use brute force matcher with knn to match key point from step 3 and find its corresponding matches in river 2.
5. Use ratio test to select point with minimum distance and satisfy a constraint of minimum ratio to be considered for match.
6. Join both these points.

Experimental Results:



Discussion:

- Key points are stable. The orientation assigns rotation invariance.
  Orientation of key point with maximum l2 norm of river1: 206.17718505859375
  Orientation of matching key point of river2: 129.88003540039062.
  So, every key point is most stable in the direction of orientation and magnitude
- The most prominent key point is detected and matched with the image taken from a different angle.

2(c) Bag of Words

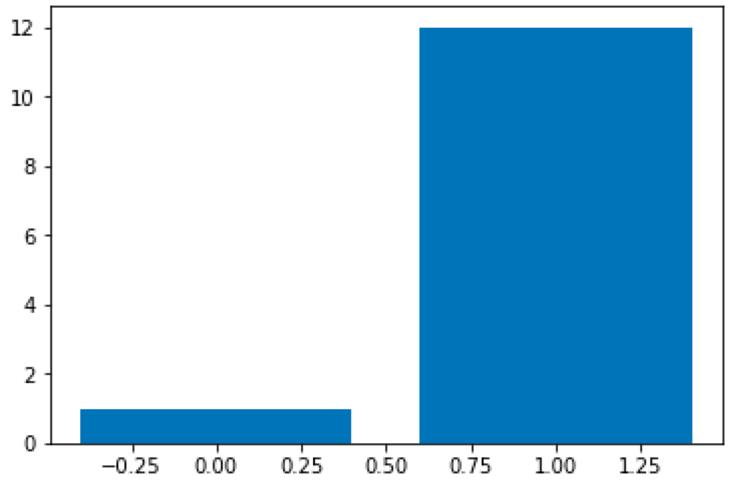
Motivation:
Bag of Visual Words is a concept inspired from Natural Language Processing. We use features from image as training data and try classifying an unknown image into some classes.

Approach:
1. Find key points and descriptors for all training samples from SIFT.
2. Concatenate all the descriptors and use them as training data.

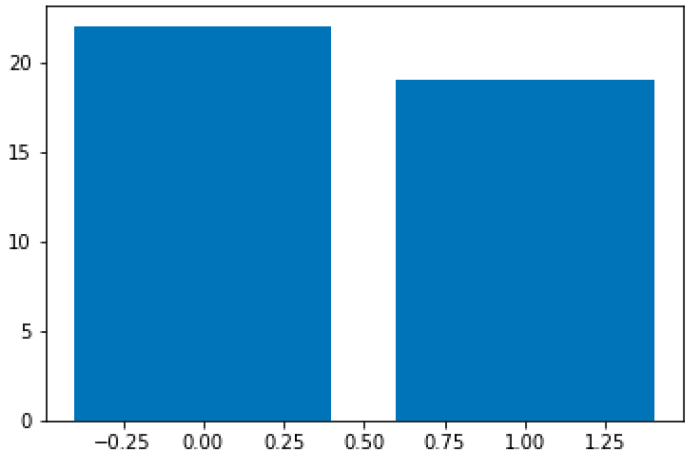
3. Use k-means clustering with 2 bin size and classify them.
4. Predict clusters of class-1 data and class-0 data.
5. Find descriptor for ‘eight’ image.
6. Predict clusters for ‘eight’ image.
7. Plot all histogram.
8. Use intersection of eight with one:
    - Find ratio of min to max of each cluster value
    - Add them all
9. Eight belongs to class with maximum intersection value.

Experimental Results:

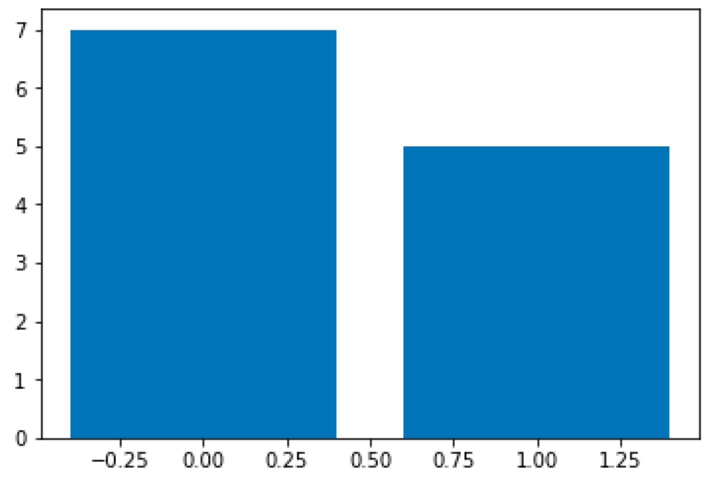
Histogram for class one:



Histogram for class zero:



Histogram of eight:



Discussion:

- We find key points and descriptors in training dataset.
- Our images are now invariable to morphological operations like rotation and scaling, due to SIFT.
- Intersection values for 0 and 1 are 0.5813397129186603 0.5595238095238095. These values change as k-means clustering is non-convex optimization.
- According to the procedure used, **8 is more similar to class 0** than class 1.

References:

http://aishack.in/tutorials/sift-scale-invariant-feature-transform-keypoint-orientation/
https://towardsdatascience.com/bag-of-visual-words-in-a-nutshell-9ceea97ce0fb
https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60
Lowe, David G. "Object recognition from local scale-invariant features." iccv. Ieee, 1999.