Homework 4: EE569

Ishaan Vasant

6989-5065-37

ivasant@usc.edu

Problem 1: Texture Analysis

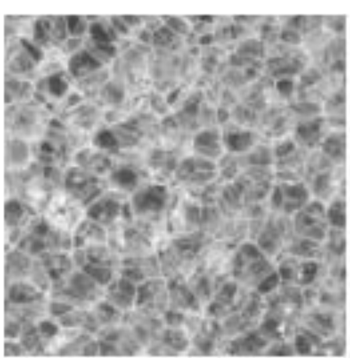
Abstract

Texture Analysis is a very important aspect of Digital Image Processing. A texture in an image can be defined as a unique pattern to a group of pixels in the image. Texture classification and texture segmentation are two processes which have been carried out in this problem.

Approach

For the first part, the 25 5x5 Laws Filters were created and applied to all the textures. The feature vectors were averaged, leading to a 25-dimensional feature vector for each image. The feature dimension was reduced from 25 to 3 using Principal Component Analysis and K-means clustering was applied to the 3 dimensions to obtain the grouping. For the second part, Laws filters were applied to the image and energy features were computed using the window approach. The energy features were then normalized with respect to the L5'L5 kernel. K-means algorithm was used to segment the image and the output was produced using 7 different gray levels. For the third part, the same procedure as the second part was carried out with the exception that PCA, dilation and filling were used to obtain the results.

Results

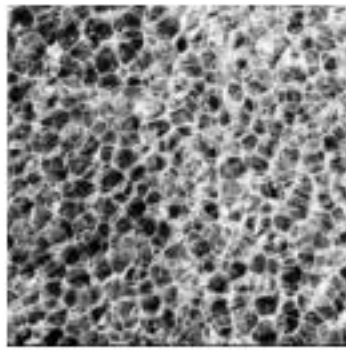


Texture 1

Texture 2

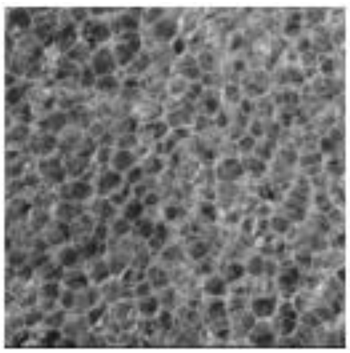
Texture 3

Texture 4



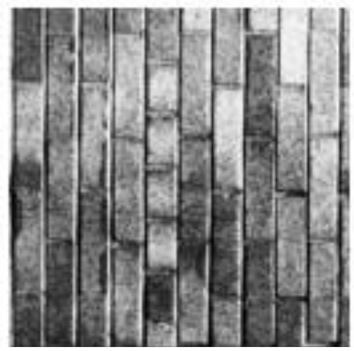
Texture 5

Texture 6



Texture 7

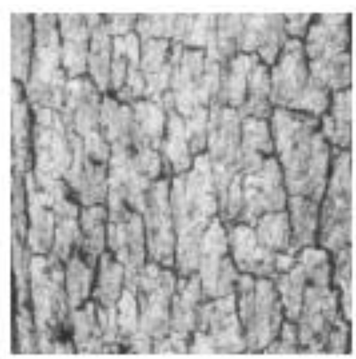
Texture 8



Texture 9

Texture 10

Texture 11



Texture 12

Fig 1: Textures 1 - 12

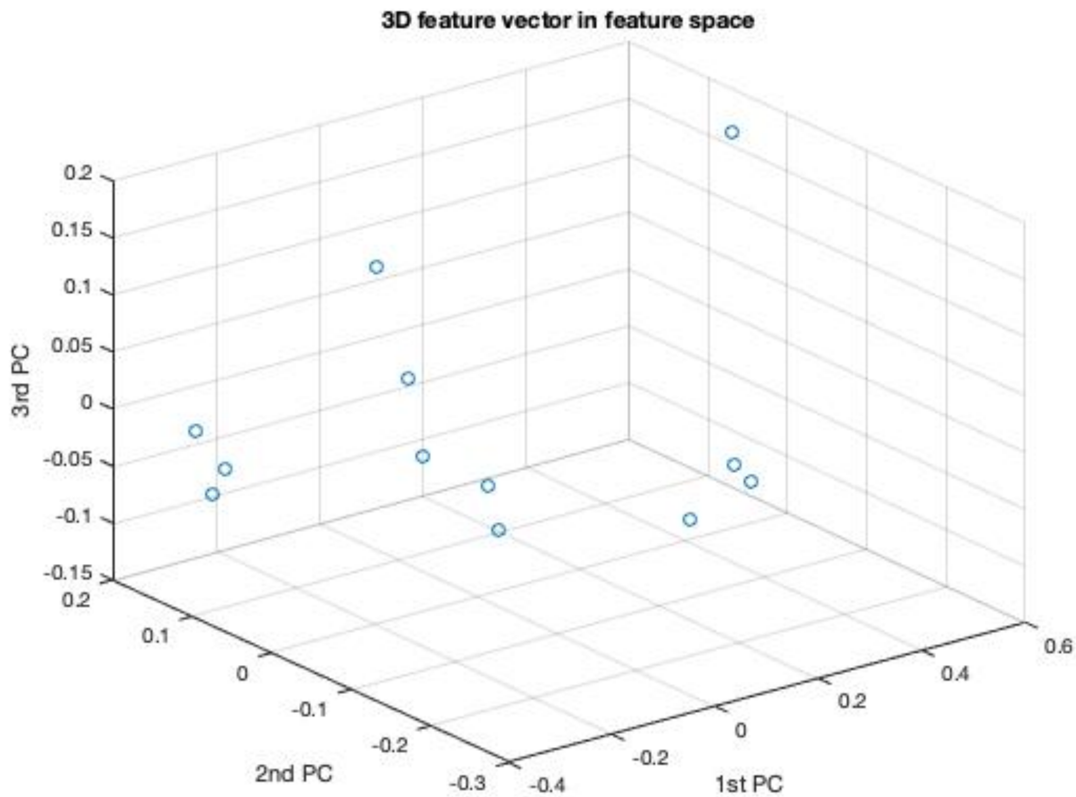


Fig 2: 3D feature vector in feature space

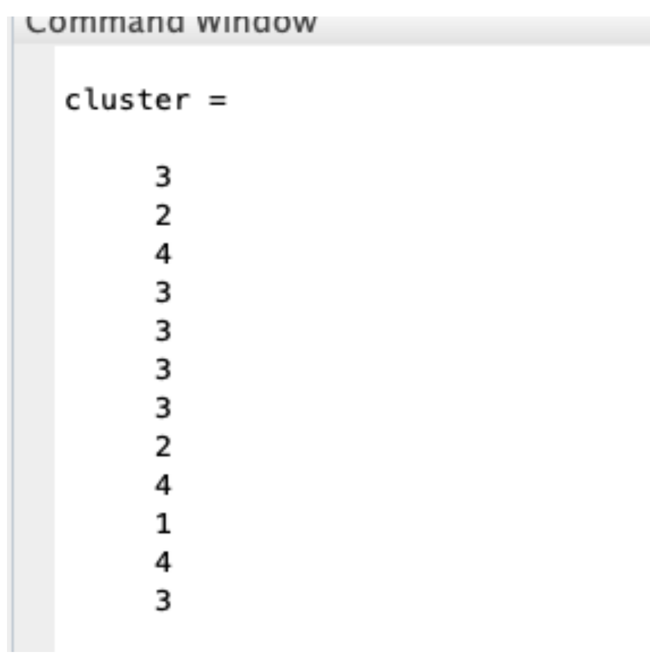


Fig 3: Clustering without feature normalization

```
>> HW4_1a

cluster =

     4
     3
     1
     2
     4
     2
     4
     3
     1
     1
     1
     2
```

Fig 4: Clustering with feature normalization and PCA

```
D =

  Columns 1 through 10

    4.6711    0.3127    0.0715    0.0947    0.3689    0.1631    0.2881    0.3524    0.3482    0.2353

  Columns 11 through 20

    0.1606    0.3614    0.3959    0.3908    0.3134    0.1809    0.3631    0.3947    0.3868    0.2996

  Columns 21 through 25

    0.1255    0.3280    0.2740    0.3058    0.1545
```

Fig 5: Mean of absolute values of normalized averaged feature vectors

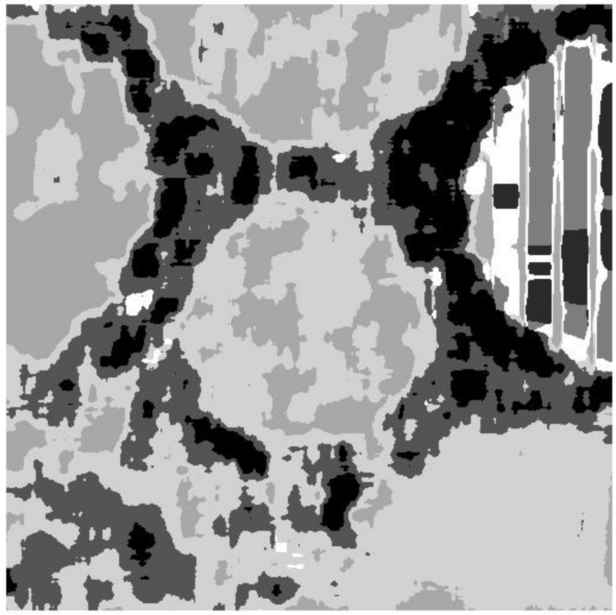


Fig 6: Segmentation using 11x11 window

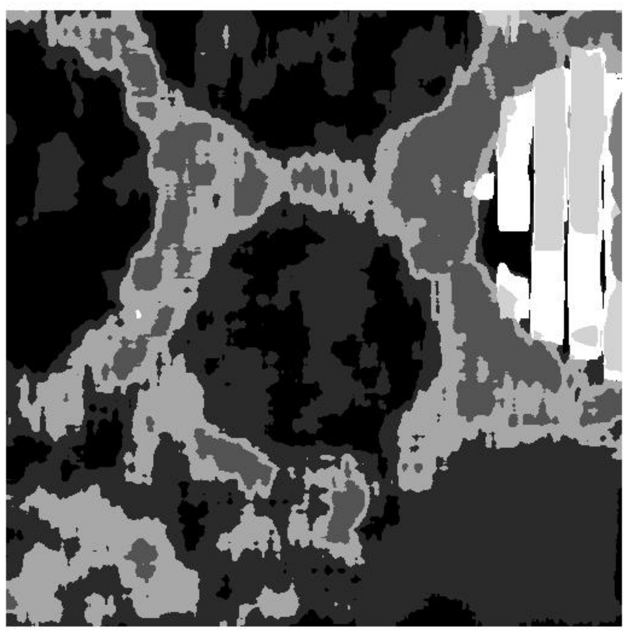


Fig 7: Segmentation using 13x13 window

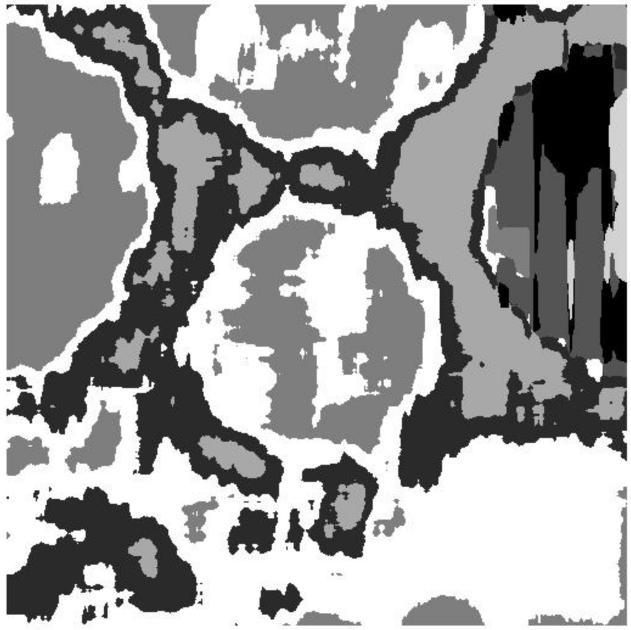


Fig 8: Segmentation using 17x17 window

Fig 9: Segmentation using 20x20 window

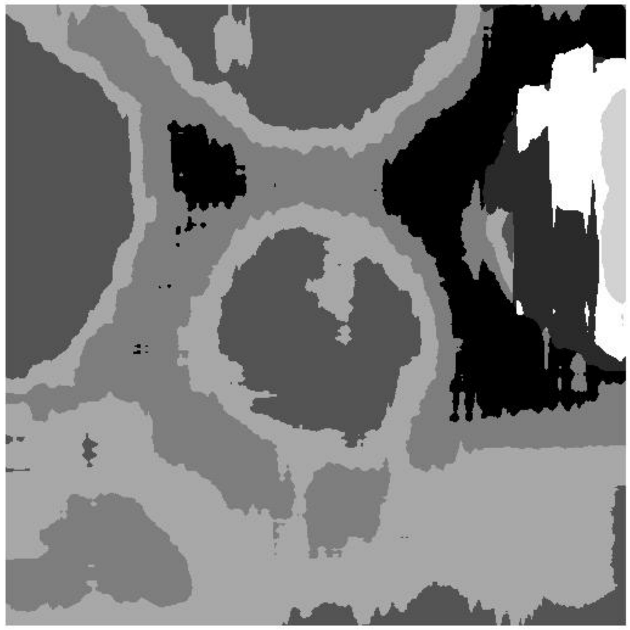


Fig 10: Segmentation using 30x30 window

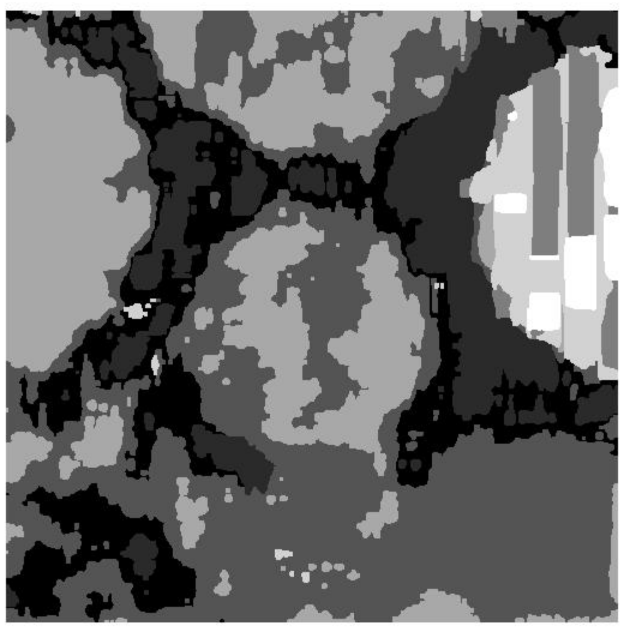


Fig 11: Segmentation using 13x13 window, PCA, dilation and filling

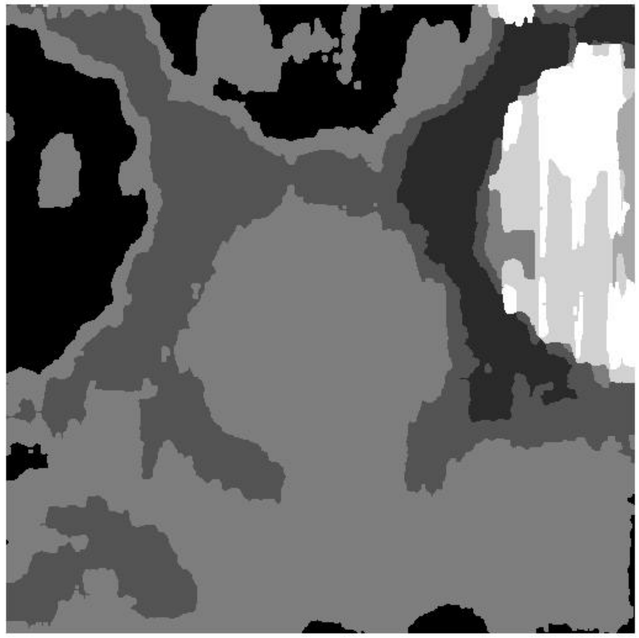


Fig 12: Segmentation using 17x17 window, PCA, dilation and filling

Discussion

a)

1. According to the mean of absolute values of normalized averaged feature vectors (seen in Fig 5), the feature with the largest absolute value is the first feature and the feature with the smallest absolute value is the third feature. Therefore, the first feature (L5’L5) has the strongest discriminative power and the third feature (L5’S5) has the weakest discriminative power.

2. On visual inspection, the correct grouping should be: -

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

   The numbers indicate the texture number.

   K-means clustering without feature normalization results in 4 misclassified points as seen in Fig 3.

K-means clustering with feature normalization results in only 1 misclassified point as seen in Fig 4 with the grouping as follows: -

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

It is seen that only texture 10 was misclassified to be in the group with 3,9 and 11 instead of with 2 and 8.

Texture 9 average feature vector	**Texture 10 average feature vector**	**Texture 8 average feature vector**
5945.45312500000	3139.40625000000	4539.85937500000
800.763183593750	511.707885742188	621.849975585938
565.038940429688	345.905517578125	368.239624023438
631.685913085938	359.186279296875	353.958740234375
1215.20837402344	721.132690429688	634.592773437500
337.903930664063	239.137084960938	610.836791992188
94.0355224609375	101.710083007813	221.077636718750
61.8697509765625	79.9843750000000	144.072998046875
70.4010009765625	98.6920166015625	146.401733398438
153.945190429688	233.993774414063	266.436889648438
175.813964843750	106.843139648438	358.552001953125
56.1229248046875	44.2570800781250	144.129272460938
37.5988769531250	36.4921875000000	100.702270507813
42.7357177734375	48.3111572265625	107.912963867188
93.9306640625000	126.406127929688	209.195312500000
178.985961914063	89.7998046875000	342.383300781250
59.1007080078125	35.3735351562500	146.591064453125
40.0913085937500	28.4593505859375	109.146728515625
45.7800292968750	38.3509521484375	124.107055664063
102.358276367188	102.643188476563	257.577758789063
376.236206054688	150.838012695313	629.044311523438
86.0467529296875	44.6094970703125	218.500732421875
125.132446289063	57.8527832031250	275.721435546875
98.7235107421875	58.6883544921875	267.180786132813
222.551879882813	151.862670898438	593.916992187500

It can be seen that the average feature vector for texture 10 is closer to the average feature vector of texture 9 than that of texture 8. Therefore, it can be understood why the K-means algorithm clustered texture 10 along with texture 9 instead of texture 8.

Interestingly, the result was exactly the same when using Principal Component Analysis for feature dimension reduction. Therefore, we can say that there was no significant impact or effect of PCA feature dimension reduction on the K-means clustering.

b) Texture segmentation was done using windows of size 11x11, 13x13, 17x17, 20x20 and 30x30. It is observed that the segmentation result got better with increasing window size. If the window size gets too big though, the result may not be as good since the different regions may merge.

c) Texture segmentation with PCA, dilation and filling was done using 13x13 and 17x17 window sizes and both show a better result when compared to the texture segmentation without PCA, dilation and filling.

Problem 2: Image Feature Extraction

Abstract

Feature extraction is a dimensionality reduction process, where an initial set of raw variables is reduced to more manageable features for processing. Image feature extractors are useful for representing the image information in a low dimensional form. Image feature extraction techniques such as Scale Invariant Feature Transform and Bag of Words have been carried out.

Approach

For the first part, the original SIFT paper was analyzed and the questions were answered in discussion. For the second part, the two river images were read. The key points of both images were found. The key-point with the largest scale in river image 1 was picked and its closest neighboring key-point in river image 2 was found. Fast Library for Approximate Nearest Neighbors (FLANN) and Brute Force (BF) techniques were used. For the third part, the images of zeros and ones were used to form a codebook. The SIFT feature vector was extracted for the image of eight and histogram of the Bag of Words was obtained.

Results

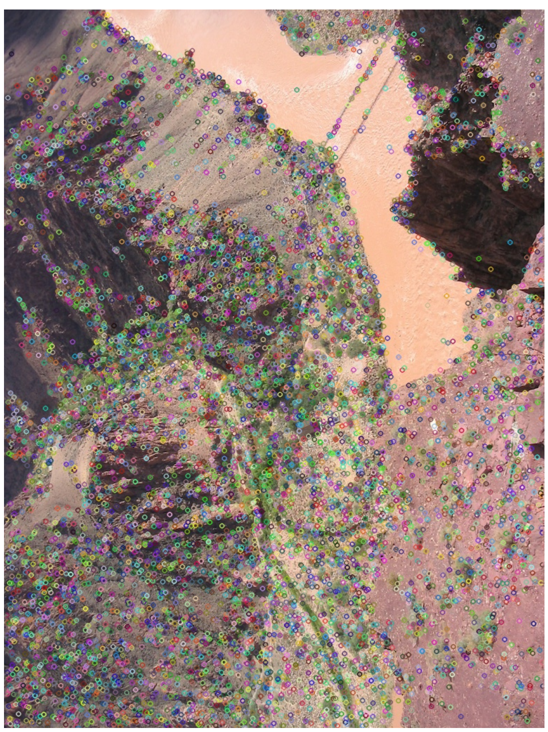


Fig 13: SIFT key points – river image 1

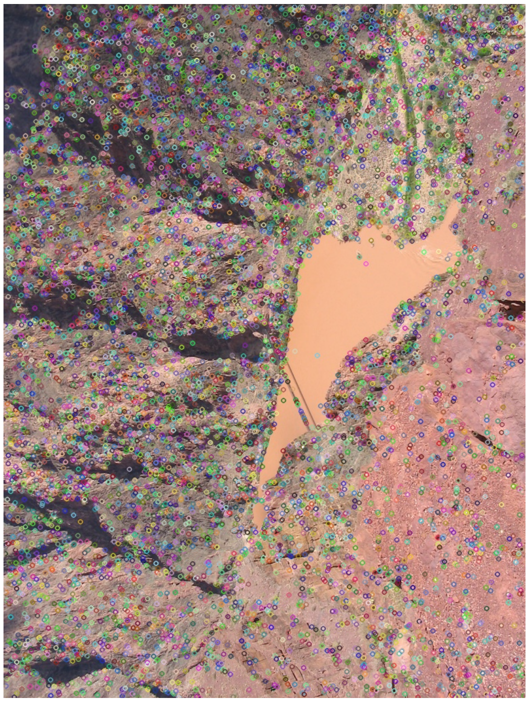


Fig 14: SIFT key points – river image 2

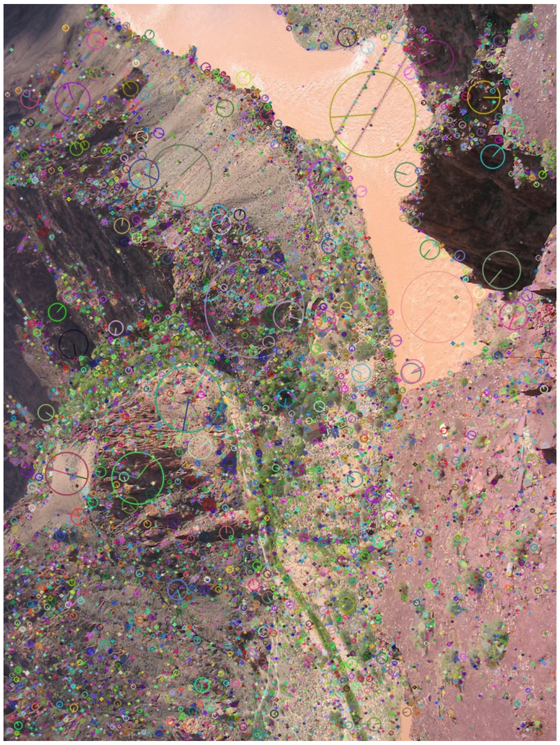


Fig 15: SIFT key points orientation – river image 1

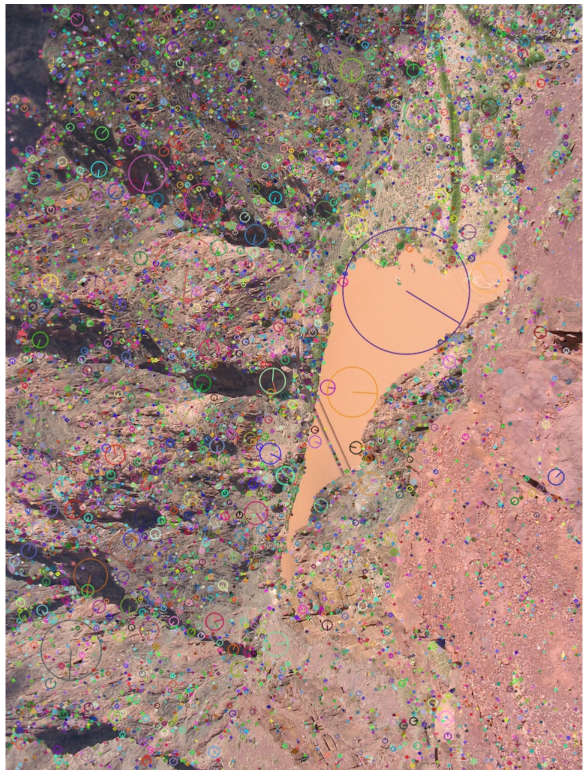


Fig 16: SIFT key points orientation – river image 2

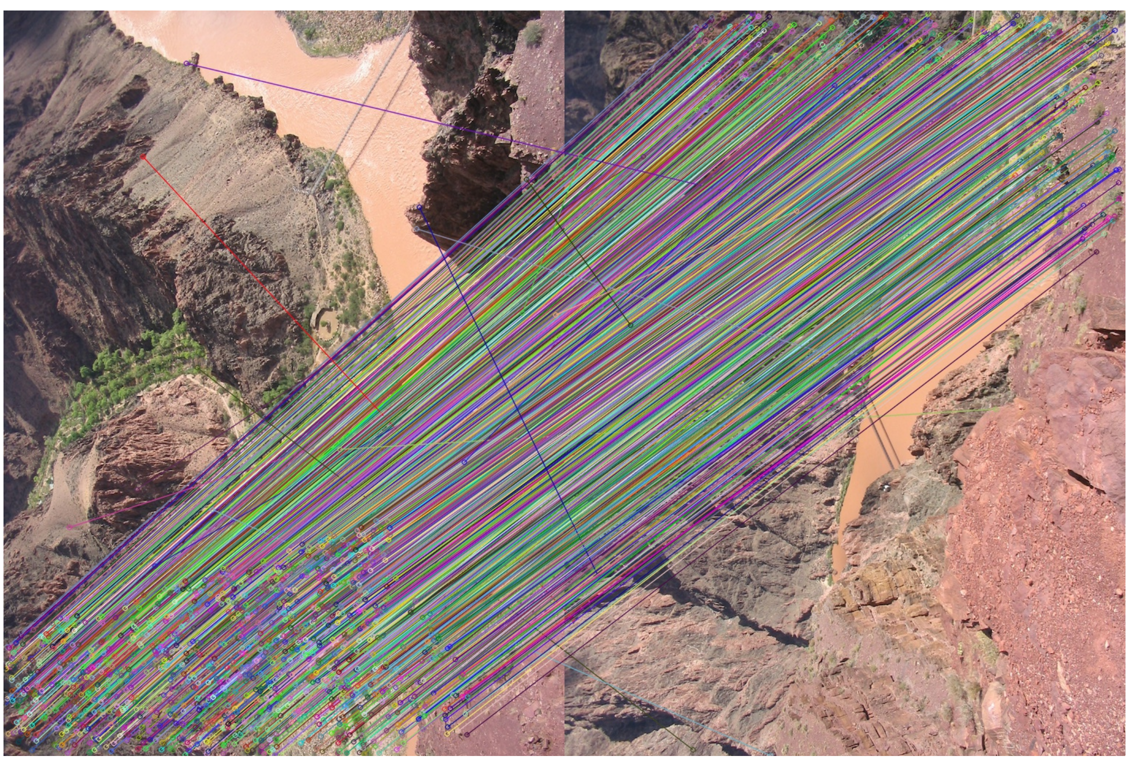


Fig 17: Matching – Brute Force

Fig 18: Largest scale key point matching – Brute Force

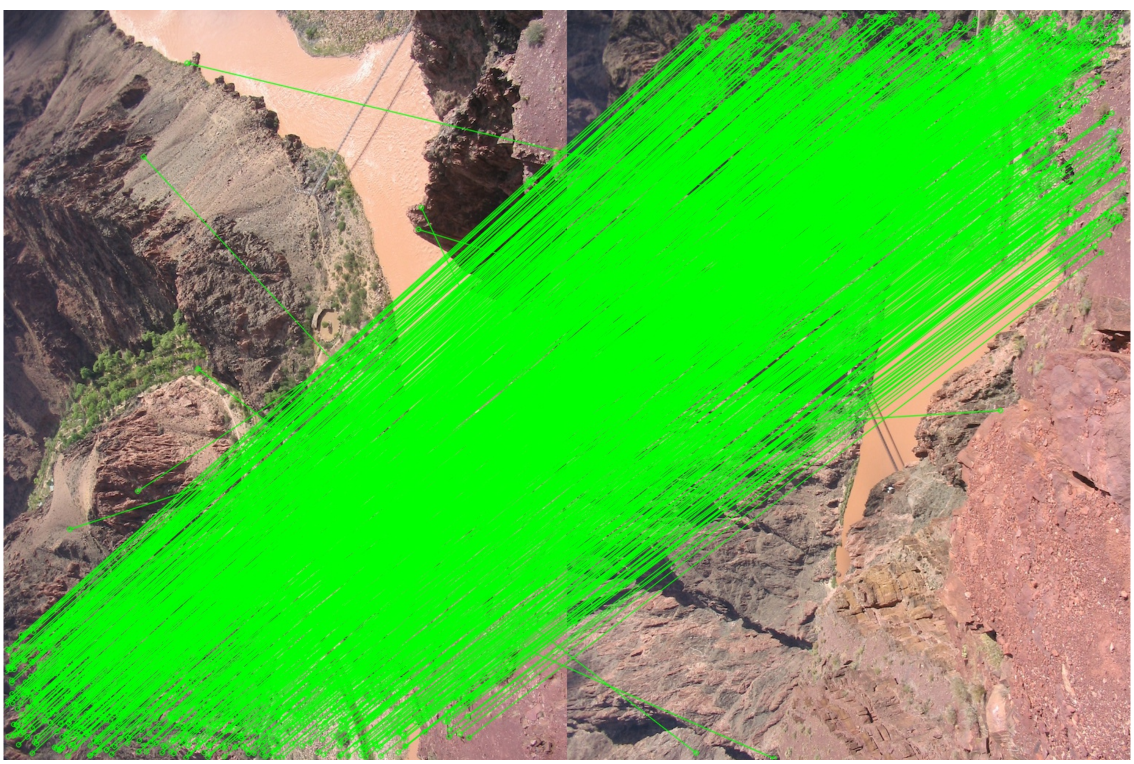


Fig 19: Matching – FLANN

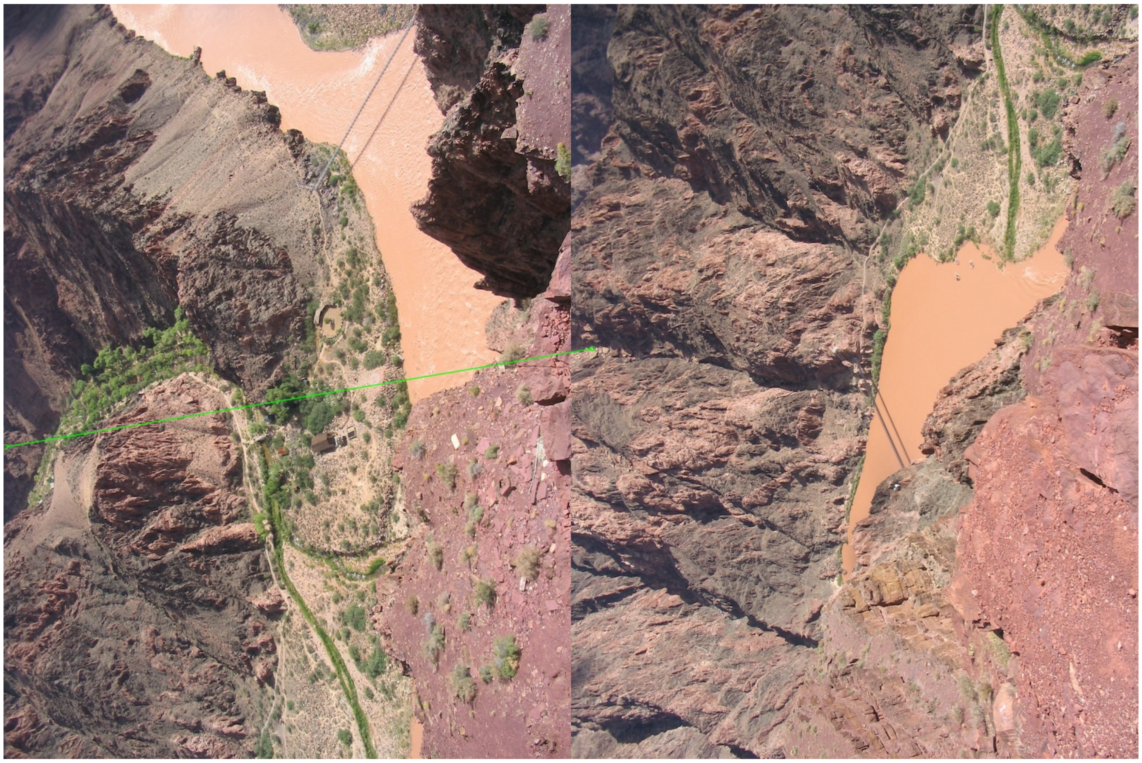


Fig 20: Largest scale key point matching – FLANN

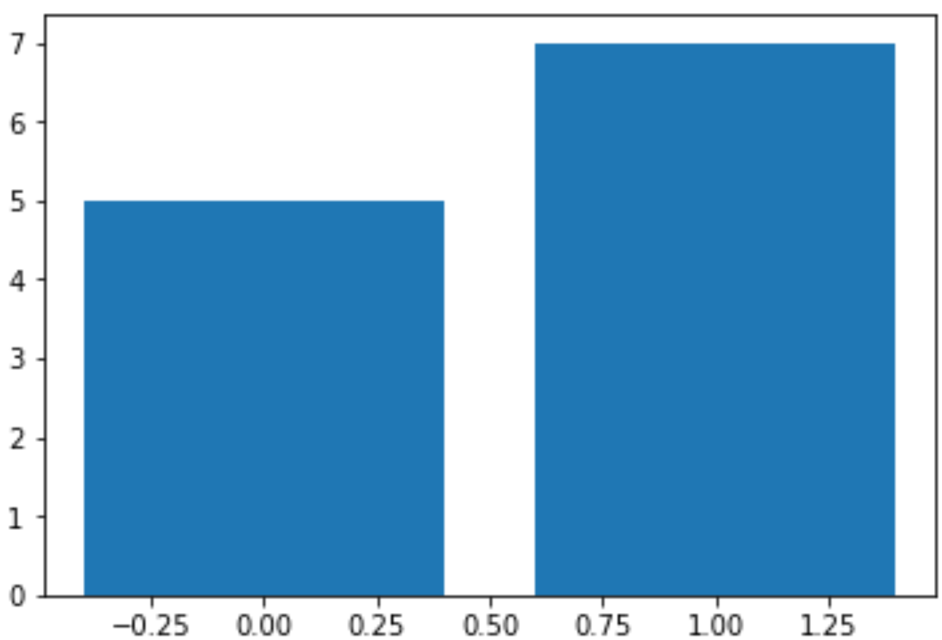


Fig 21: Bag of Words Histogram for image of number eight

Discussion

a)

1. From the paper abstract, the SIFT is robust to what geometric modifications?

   SIFT is robust to scaling, rotation and translation.

2. How does SIFT achieves its robustness to each of them?

   SIFT is robust to scaling through multi-scale filtering. It is accomplished by searching for stable features in the image across all possible scales. Since Difference of Gaussian is a close estimate to the scale-normalized Laplacian of Gaussian, normalization of the Laplacian with the factor 2 accomplishes accurate scale invariance.

   SIFT is rotation and translation invariant through locating key-points as extrema of DoG function where each key-point is assigned a canonical orientation.

   SIFT is robust to rotation by assigning a steady orientation to each key point of the image based on the local image properties. The key point descriptors can be represented in relation to this steady orientation and thus are not disturbed.

   SIFT is robust to translation due to the normalization of the key point descriptors that permits a significant shift in gradient positions by generating orientation histograms over 4x4 sample.

3. How does SIFT enhances its robustness to illumination change?

   SIFT is robust to illumination change since the vector is normalized to unit length. Vector normalization negates the change in image contrast since it is just each pixel multiplied by a constant hence multiplying the gradient by the same constant. Change in brightness is just addition of a constant to each image pixel doesn't affect gradient values because they are calculated from the pixel differences. Non-linear Illumination changes cause a sizeable change in relative magnitudes for gradients but less likely to affect gradient orientations. This is handled by reducing the influence of large gradient magnitudes. This is done by thresholding the values in the unit feature vector to be no larger than 0.2. Then it is re-normalized to unit length. Illumination robustness is achieved by thresholding the gradient magnitudes at a value of 0.1 times the maximum possible gradient value. Canonical orientation is determined by the peak in a histogram of local image gradient orientation.

4. What are the advantages that SIFT uses difference of Gaussians (DoG) instead of Laplacian of Gaussians (LoG)?

   The main advantage of Difference of Gaussians (DoG) over Laplacian of Gaussians (LoG) is speed.

5. What is the SIFT's output vector size in its original paper?

   Vector size = 8x4x4 + 8x2x2 = 160

b) The key points of both images were found. The key-point with the largest scale in river image 1 was picked and its closest neighboring key-point in river image 2 was found. Fast Library for Approximate Nearest Neighbors (FLANN) and Brute Force (BF) techniques were used. The final matching for the largest scale key point was done using both techniques and it is seen that the FLANN and BF techniques do not yield the same results.

c) The images of zeros and ones were used to form a codebook. The SIFT feature vector was extracted for the image of eight and histogram of the Bag of Words was obtained. The histogram shows the count for 0 to be equal to 5 and the count for 1 to be equal to 7. This is interesting since one would think the number 8 would be more similar to the number 0 than the number 1.