Salient Point Descriptors and Image Matching

(Implementation for this problem is done with C++ using OpenCV 2.4)

(a) Extraction and Description of Salient Points

Introduction

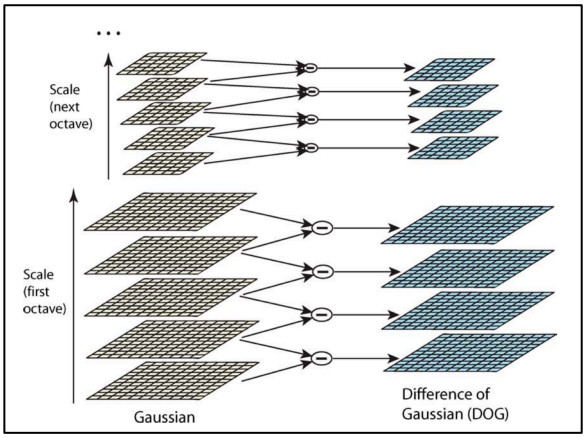
In this problem, we are asked to implement SIFT and SURF algorithms to two image to extract keypoints and their descriptors.

SIFT Algorithm Overview:

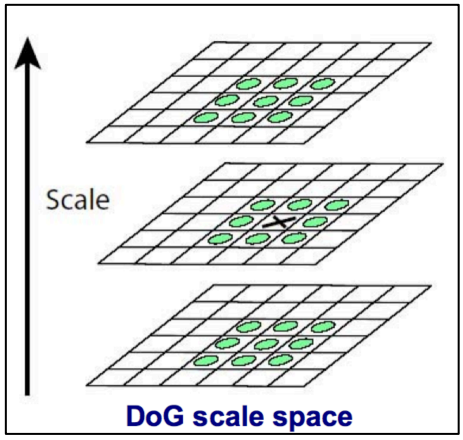
SIFT (Scale-invariant feature Transform) is an algorithm to detect and describe local features in images. We can explain the overview of this algorithm step by step:

1)Scale-space extrema detection: Keypoints are detected using scale-space extrema in difference-of-Gaussian function D. The purpose in this is to approximate Laplacian operator using Difference-of-Gaussian.

$$D(x,y,\sigma) = (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y)$$



2)Key point selection and localization



A key point is selected only if it is a local minimum or maximum in the 3D DoG (Difference of Gaussian) space.

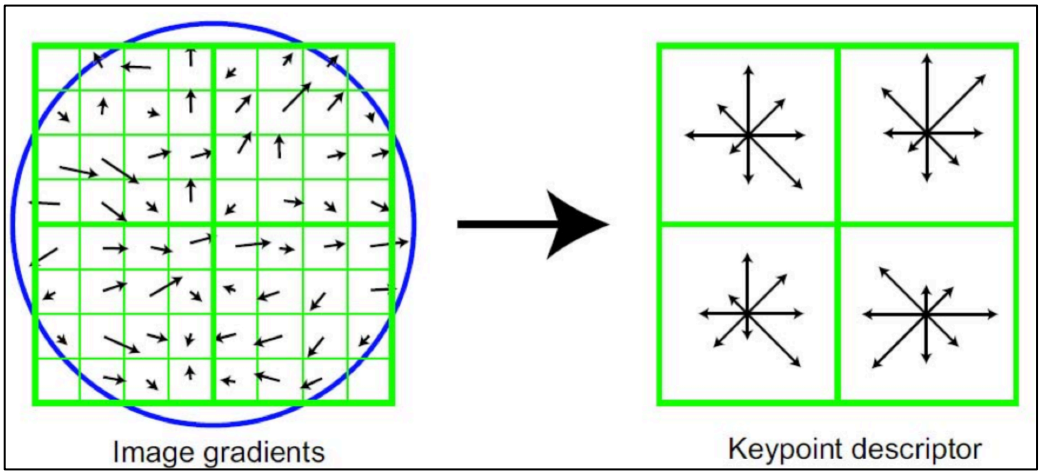
After that keypoints are localized using mathematical procedure.

3)Keypoint orientation assignment: Each keypoint is assigned an orientation to achieve invariance to image rotation. Gradient magnitude and direction is calculated around the keypoint.

4)Keypoint descriptor: A 16x16 neighbor around the keypoint is taken. This structure is divided into 16 sub-blocks (4x4). In each sub-block, a histogram of 8 bins in which each bin represents a different direction is created.

As a result, we have 16 sub-blocks and each sub-block has 8 bins, therefore we have 16 x 8 = 128 dimensional feature vector for each keypoint. This feature vector is a descriptor of the keypoint.

The figure below, illustrates the concept described above, in the figure below we have 4 (=2x2) sublocks, however in the algorithm there are 16 sub-blocks.



SURF Algorithm Overview:

Speeded Up Robust Features (SURF) algorithm is a local feature detector and descriptor. SURF is similar to SIFT in the sense that they share the basic principles in the steps of algorithm and it gives similar results however there are some differences. In the next section, we will observe the results of SIFT and SURF.

Results:

Bus Image:

SIFT Algorithm (Bus)

SURF Algorithm(Bus)

To compare the performance of SIFT and SURF, I will give the number of keypoints and the time it took to compute those keypoints.

Result Table for the Bus Image:

	Number Of Keypoints	**Time for Computation (in seconds)**
SIFT	778	0.109974
SURF	1688	0.0644897

Sedan Image:

SIFT Algorithm

SURF Algorithm

Result Table for the Sedan Image:

	Number Of Keypoints	**Time for Computation (in seconds)**
SIFT	284	0.0493211
SURF	837	0.0344572

As we can see, SURF algorithm extracts more keypoints and does the computation faster than SIFT algorithm.

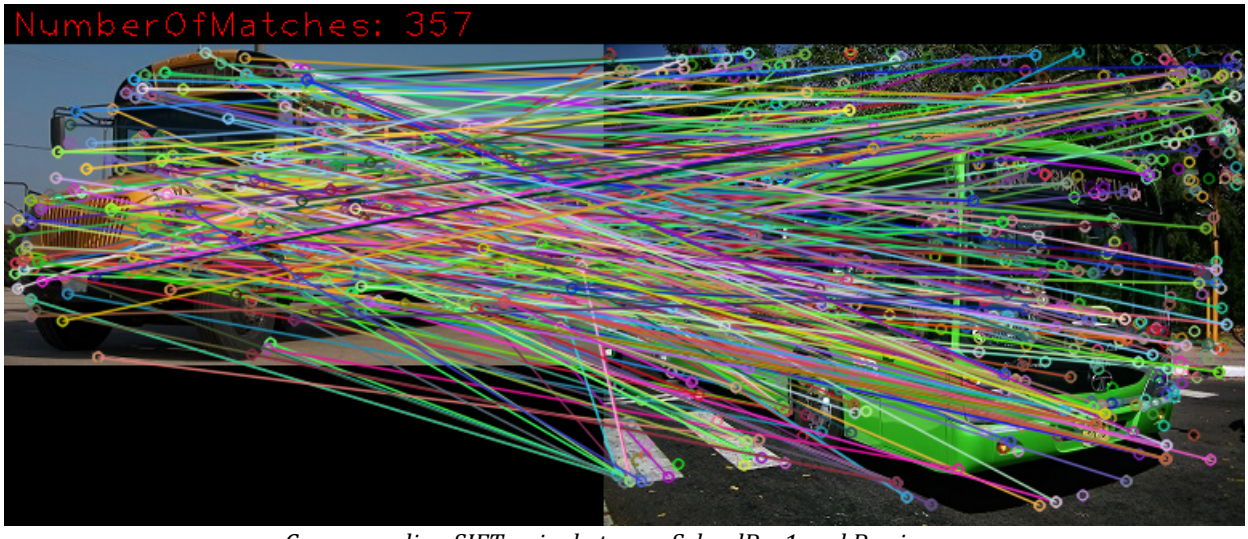
(b) Image Matching

SIFT features are extracted and shown for two school bus images:

SIFT features extracted

Corresponding SIFT pairs

Image Pairing between SchoolBus1 and Bus:



Corresponding SIFT pairs between SchoolBus1 and Bus image

Image Pairing between SchoolBus1 and Sedan:

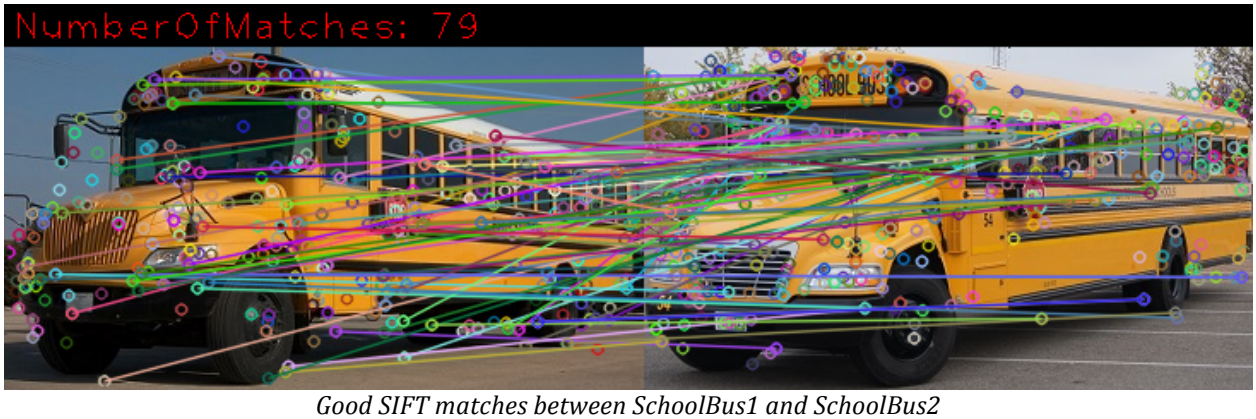
Corresponding SIFT pairs between SchoolBus1 and Sedan image

As we can see from the results above, the SIFT matching between SchoolBus1 and SchoolBus2 gives somewhat good results, however as they are the same object with different camera angles, there are some invalid matches.

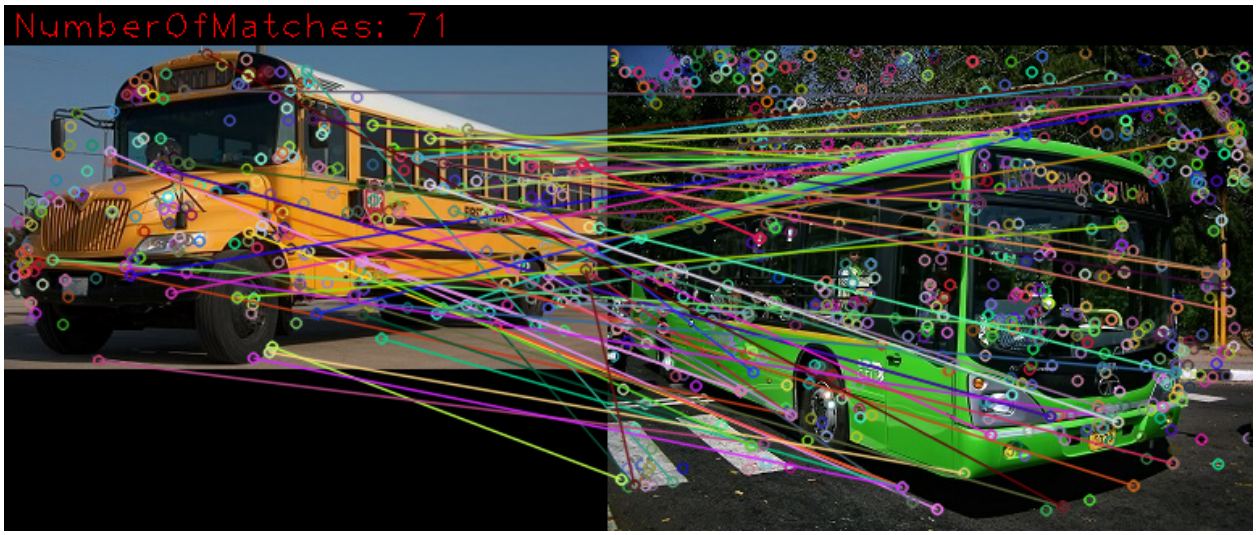
The matches between SchoolBus1-Bus and SchoolBus1-Sedan give invalid matches because of the fact that they are not the same objects, and SIFT matches are highly effected by the background information.

We can eliminate some of the weak matches and be left with some of the strong matches. We can accomplish this with a threshold and comparing this with how strong a match pair is, the weak ones are going to be eliminated. We are left with good-strong matches in the figures below:

Good Matches:



Good SIFT matches between SchoolBus1 and SchoolBus2



Good SIFT matches between SchoolBus1 and Bus

Good SIFT matches between SchoolBus1 and Sedan

As we can see, there are still some invalid matches and results are effected by background information.

(c) Bag of Words

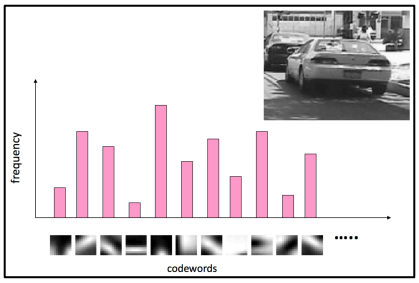
Introduction:

We can explain Bag of Words method by one of its purposes which is 'Document Classification' . In this method documents (which have contain many words) are represented as frequencies of words from a dictionary.

Words \ Documents	Doc-1	Doc-2	Doc-3	Doc-4
Bank	1	1	0	0
Loan	1	0	0	3
Water	0	2	2	1
Farmer	0	0	0	1

The image on the right shows us that different documents can be represented as a histogram of words, and this histogram can be used as a feature vector for classification.

The same logic also applies in the images as well. If we can come up with visual 'words', we can represent images as the frequencies of those words.



The methodology for creating the visual words involves, SIFT feature extraction and k-means clustering. In the next section, we will explain this methodology.

Methodology:

In the first step, we extract SIFT features from the three images (Bus, Sedan, and SchoolBus1). Each feature vector has dimension 128. In the figure below, we see the SIFT feature keypoints of the three images.

In the second step, we put all these features from all three images into the same feature matrix and perform k-means clustering in this matrix. We are asked to create K=8 bins, therefore we should perform k-means clustering with the parameter k=8 which will create 8 clusters.

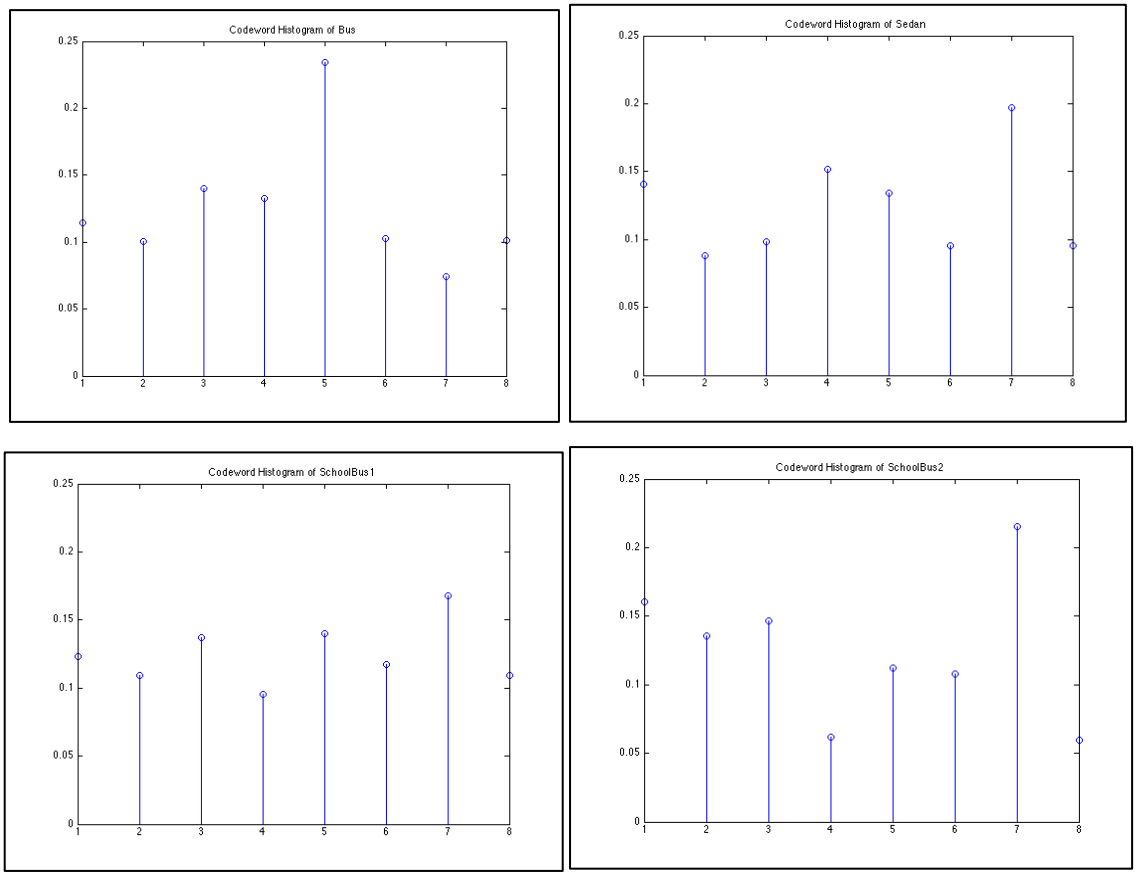
Each 8 cluster mean will be used as the 'visual word'. In other words, by obtaining these visual words, we can represent an image with the frequencies of those visual words and predict how similar they are by looking at the frequency histogram.

Each visual word(codeword) has dimensionality 128, and we have 8 codewords. Now in the next step, we need to represent each image with those codeword. Each SIFT feature vector in the images is quantized to its closest codeword, which is derived from k-means. Then we just count the number of occurances of those codewords in each image and create the representation of the image with the 8 codewords.This representation is nothing but a histogram of those codewords.

In my program, I normalized each images histogram so that each element in the histogram is between 0 and 1.

Results:

In the figures below, we see the histograms of images.



In this problem, we are asked to match SchoolBus2's codeword histogram with other images'. To measure how close these histogram 8 dimensional feature vectors I am using Euclidian distance metric. The distances between these histograms are given below.

	Distance
Between SchoolBus2 and SchoolBus1	0.0941003
Between SchoolBus2 and Bus	0.211704
Between SchoolBus2 and Sedan	0.123168

As we can see from the results above, SchoolBus2 is closest to SchoolBus1, which is an expected result. However, SchoolBus2 is closer to Sedan image rather than the Bus images.

I also wanted to apply PCA to reduce the dimensionality of vectors from 128 to 20, in goal of improving the results and to see if the results will get effected. I applied PCA to the feature matrix, which contains all the SIFT feature vectors for all of the images. After applying PCA and following the same procedure, I got these results:

With PCA Applied	Distance
Between SchoolBus2 and SchoolBus1	0.0903105
Between SchoolBus2 and Bus	0.180379
Between SchoolBus2 and Sedan	0.125227

As we can see, applying PCA does not seem to change the results that much.

We will discuss this in the next section.

Discussion:

If we look at the SIFT keypoints in Bus and Sedan images, they seem to get effected from the background information. That might be causing mismatches and invalid results in SIFT matching and classifications.