Visual Perception SIFT Lab Report

Gourab Ghosh Roy and Mohammad Rami Koujan

21st May 2016

1 Introduction

One of the essential aspects in Computer Vision is performing image matching, object or scene recognition, 3D structure from multiple images, and motion tracking. Achieving these critical tasks, in fact, requires the use of good features in an image. This raises two fundamental questions. The first one is what kinds of features should be detected and the second one is how to describe those features. Good features are those which are repeatable (should be detected despite changes in viewing conditions), distinctive (the feature descriptors should permit a high detection rate and low false positive rate), compact, and efficient. Two main approaches are usually used to find feature points and their correspondences. The first is to find features in one image that can be accurately tracked using a local search technique such as correlation or least square. The second is to independently detect features in all the images under consideration and then match features based on their local appearance. The former approach is more suitable when images are taken from nearby viewpoints or in rapid succession (e.g., video sequences), while the latter is more suitable when a large amount of motion or appearance change is expected. Scale Invariant Feature Transform (SIFT), which is the main focus of this lab assignment, is a method proposed by Lowe which transforms image data into scale-invariant coordinates relative to local features [1]. The features are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination [1].

2 Generating the testing dataset

In this section, three image sequences are generated so as to evaluate the performance of the SIFT descriptor. Those sequences have the following structure:

1. A reference image which is a synthetic underwater scene.
2. A set of images describing different camera movements around the same region of interest of the scene.

The first sequence consists of a set of 16 underwater images resulting from tilting the camera left, right, up, and down at four different angles around the scene axes. In order to construct this sequence, the following steps are followed:

1. A window of size 750x500 that contains an interesting scene is cropped from the original image. This cropped image represents the reference image.
2. Another two windows which are centered around the window of step 1, are cropped ($Iorig\_w$ and $Iorig\_w2$). Those two windows have sizes of 500x950 and 750x750 respectively. The first one is used to build the images that result from titling the camera left and right, and the second one is used for up and down titling cases.
3. Tilting degree is defined by a parameter k in terms of pixels, where it is multiplied by 1,2,3,4 to represent the four different angles of tilting for each direction.
4. A function called *fitgeotrans* is used to fit geometric transformation to control point pairs defined by image $Iorig\_w$ or $Iorig\_w2$ (depending on the direction of tilting) and the tilted image which is represented by four corners, two of them are shifted according to the tilting direction.

5. The fitted geometric transformation of the preceding step, which is obtained in a form of a transformation matrix, is used by an image transformation function *imwarp* to generate the transformed image.

6. The homography matrix of each image is set with the transformation matrix of step 4.

7. Finally the resultant transformed image is cropped around its center with a window of size 750X500.

Figure 1 shows an example of a reference and projectively transformed images from the created sequence.

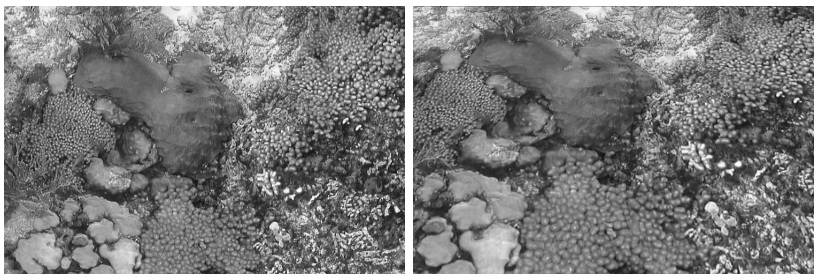


Figure 1: Reference (left) and transformed image (right) that is the result of tilting the camera up

The second sequence, zoom sequence, contains a set of 9 images that represent different zooms of the original scene, from 110 % up to a 150 % in increments of 5 %. This sequence is built by first creating a multidimensional spatial transformation structure (called a TFORM structure) using *maketform* function. This structure is used by the *imtransform* function which is responsible for creating the transformed image. The *maketform* function takes the type of transformation, which is in this case affine, and the transformation matrix as input parameters where the following matrix is the transformation matrix.

$$T = \begin{bmatrix} zoom(i) & 0 & 0 \\ 0 & zoom(i) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The "zoom" parameter is, in fact, an array that contains the distinct necessary zoom values for this sequence. The homography matrix is created by the multiplication of three matrices. The first one from the right is a translation matrix, the purpose of which is to translate the origin of the image from the top left corner to the center since the rotation is done around the center. The second matrix is T matrix defined previously and the third one is another translation matrix to set the top left corner again as the origin of the image. Figure 2 shows the original image and the transformed image with a zoom value of 1.45 %.

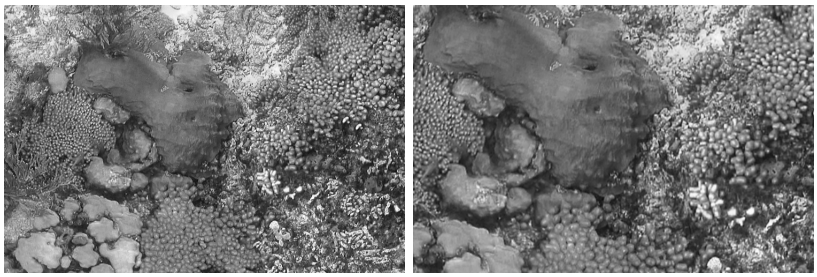


Figure 2: Reference and zoomed image by 1.45 %

The third sequence, rotation sequence, has a set of 18 images that are the result of rotating the camera around the center of the scene by angles ranging from -45 to 45 degrees. Constructing this sequence is done as follows:

1. A window of size 1500X1500 centered around the reference image is cropped from the original underwater scene image.

2. An array, theta, is defined to hold the different necessary rotation angles values.

3. The *maketform* function is used to create a spatial transformation structure using affine transformation type and the following rotation matrix:

$$T = \begin{bmatrix} cos(theta(i)) & -sin(theta(i)) & 0 \\ sin(theta(i)) & cos(theta(i)) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

4. The homography matrix is created as in zoom sequence except that the transformation matrix here is the same as the rotation matrix instead of the scale matrix.

5. An image transformation function *imwarp* is used to transform the reference image using the previously created structure.

6. The transformed image of the previous step is cropped around the center producing another image of size 750X500.

Figure 3 shows an example of a pair of a reference and rotated images from the rotation sequence.

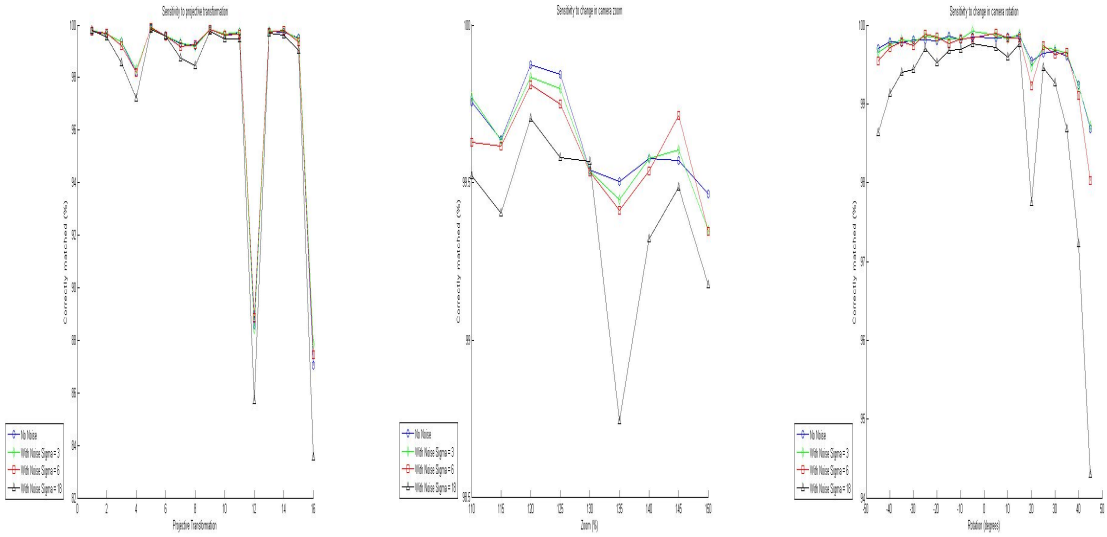
Figure 3: Reference image (left) and rotated image (right) by 45 degrees

3 Results

3.1 Testing the performance of SIFT descriptor

In this section, David Lowe's MATLAB implementation of SIFT [1,2] was used to test the performance of the SIFT descriptor. The MATLAB script *lowesifttest.m* can be used to obtain the plots of the percentages of correct matches for different transformations on the control images generated as described in the previous sections. We modified the existing function *match.m* to return not just the number of matches but also the location of those matches. The keypoint matches from the SIFT descriptor are thus obtained and they are checked for correctness by comparing with results from the homography matrices for corresponding points. An important parameter for this comparison is the threshold which controls the allowed absolute difference between the two results.

Figure 4 shows the plots for performance tests of the SIFT descriptor. One subfigure has 4 plots for the 4 different noise levels.



(a) Projective (b) Zooming (c) Rotation

Figure 4: Testing the performance of SIFT descriptor

Figure 4a shows the percentage of correct matches against projective transformations. Studying this line graph in more detail, it can be seen that titling the camera by higher angle in any of the four studied directions (up,down,left,right respectively) causes the percentage of correct matches to dip from around 99 percent for the smallest tilting angle to as low as 84 percent for right direction. Nevertheless, the amount of the shown decline is not similar for all directions which can be contributed to by the partial invariance of the SIFT features with changing of the 3D view point. In other words, tilting the camera in different directions by the same angle may not always exhibit the same behaviour. Another noticeable trend in this figure is that adding higher noise levels results in degrading the percentage of correct matches more, which for small values of σ is quite similar to the non-noisy case . Overall, there appears to be a partially invariant tendency of the SIFT features with respect to the changing in the 3D view point.

Figure 4b shows the percentage of correct matches against the increase in zooming level. The percentage does decrease a little as zooming increases but the overall trend with such low variations affirms the scale invariant nature of SIFT descriptor. The match accuracy does not decrease uniformly with increase in zoom. This can be attributed to the SIFT detection process of using scale space pyramids resulting in high matching accuracy for some levels of zoom compared to their adjacent levels. But these are very small variations. As expected the match percentage is normally higher for the case with no noise. In one or two instances the percentage is higher which might be because addition of noise (with low standard deviation) can lead to high frequency accentuation slightly improving the detection process. But the general trend with increase in noise is evident when comparing the cases of no noise and maximum noise, when the percentage of matches decreases in the latter.

Figure 4c shows the percentage of correct matches with the change in camera rotation. The percentage does not change too much with the change in rotation, the values are normally between percentages of 98-100, except the case with maximum noise. SIFT matching is better for image with no noise, but it degrades a little in presence of noise as seen in the figure. The change with rotation angle is not symmetrical in practice, because in rotating the camera in different directions the scene changes and with it the keypoints used for computing correct match percentage changes. The overall trend confirms another behavior of the SIFT descriptor namely the rotation invariance.

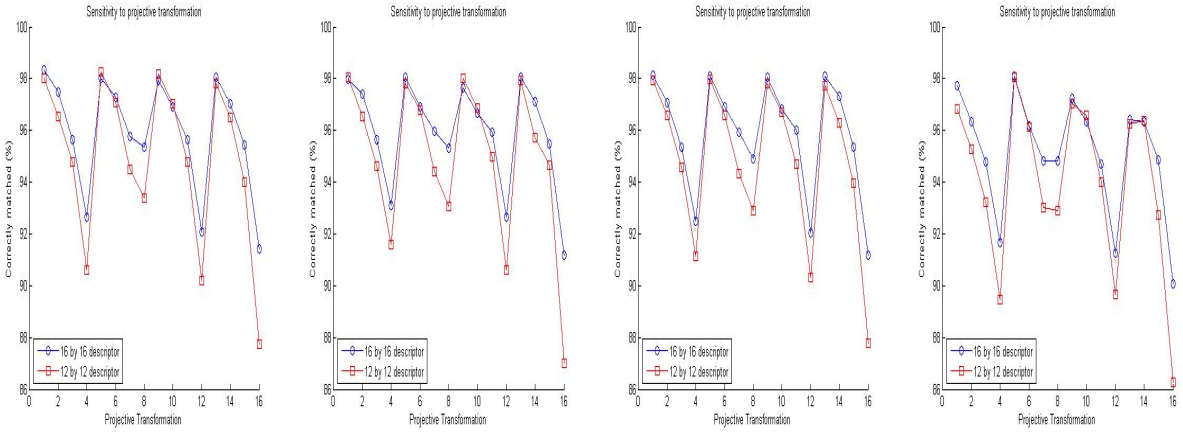
3.2 Implementing a modified SIFT descriptor

In this section, we have used the SIFT implementation in VLFeat Open Source library [3] for the assigned task of Team A. The MATLAB script *vlsifttest.m* produces the match percentage comparison plots. It calls the function *vlmatch.m* which uses VLFeat functionalities like *vl_sift* and *vl_ubcmatch*. The criteria of matching used in Lowe's implementation (angle based) and the *vl_ubcmatch* function (distance based) are different, which explains the difference in the match percentage values between results of last section and those of this section. However the trend of the graphs as presented below corroborate all observations and conclusions we have derived before.

The task at hand is to change the descriptor and subwindow sizes and test the performance of the SIFT descriptor. The parameter 'Magnif' controls the descriptor window size, as the scale of the parameter is multiplied by this factor to get the width of the spatial bin. A default value of 3.0 corresponds to the 16 by 16 descriptor with a 4 by 4 subwindowing according to the formula used in the *sift* code for computing circular window size W.

$$W = floor(sqrt(2.0) * magnif * sigma * (NBP + 1)/2.0 + 0.5)$$

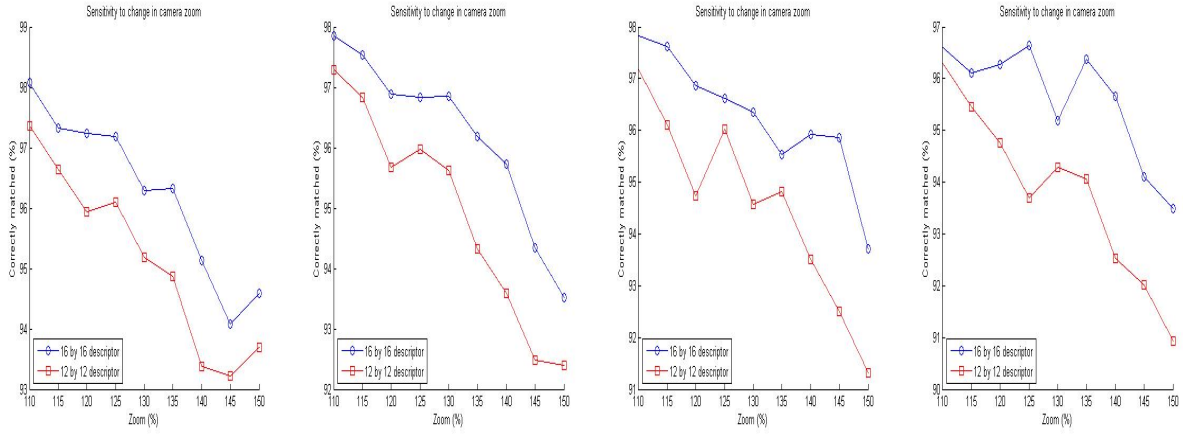
where NBP (the number of spatial bins) has a default value of 4. Keeping the number of bins same, a value of 2.2 for the 'Magnif' parameter approximately corresponds to a 12 by 12 descriptor with a 3 by 3 subwindowing.



(a) Noise $\sigma = 0$ (b) Noise $\sigma = 3$ (c) Noise $\sigma = 6$ (d) Noise $\sigma = 18$

Figure 5: Testing Modified SIFT Descriptor for Projective Transformation

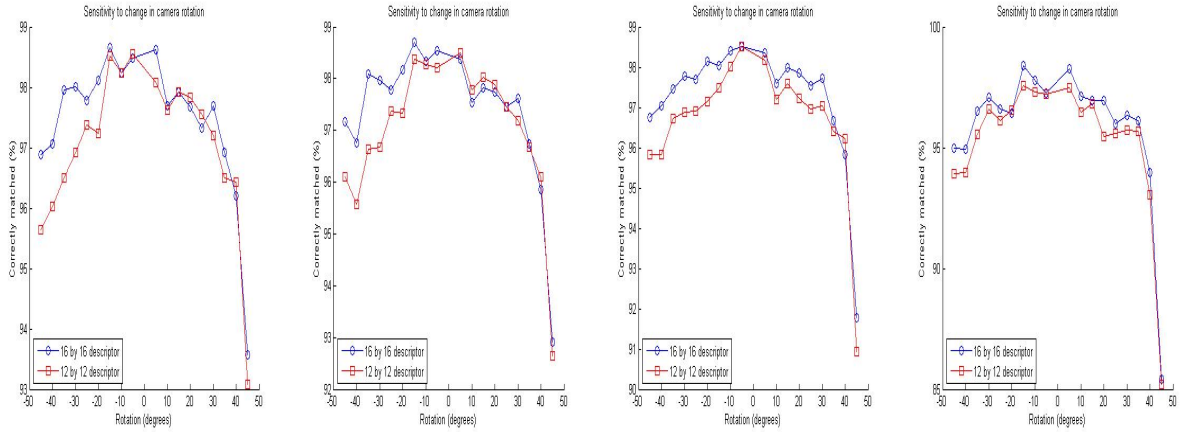
Figure 5 shows the performance comparison of the descriptors with different sizes on control images with projective transformation. Viewed together, the 16 by 16 descriptor always outperforms the 12 by 12 one. In addition, the shown values demonstrate alike repercussions of both descriptors for noise.



(a) Noise $\sigma = 0$ (b) Noise $\sigma = 3$ (c) Noise $\sigma = 6$ (d) Noise $\sigma = 18$

Figure 6: Testing Modified SIFT Descriptor for Zoom

Figure 6 shows the comparison of the descriptors with different sizes on images with different zoom levels. The default 16 by 16 descriptor always performs better than the 12 by 12 descriptor, and this difference is more pronounced as the zoom level increases. The said difference also increases with the increase in noise level.



(a) Noise $\sigma = 0$ (b) Noise $\sigma = 3$ (c) Noise $\sigma = 6$ (d) Noise $\sigma = 18$

Figure 7: Testing Modified SIFT Descriptor for Rotation

Figure 7 shows the comparison of the descriptor types for images with different angles of rotation. In this case too the 16 by 16 descriptor gives better match percentages than the 12 by 12 descriptor, except in some instances for the case without any noise for low angles of rotation. This shows that the performance of the default 16 by 16 window size is better in presence of noise because information around keypoints is lost by reducing the window size.

Considering the results obtained from all 3 control sequences, it can be concluded that the 16 by 16 descriptor with 4 by 4 subwindowing performs better than the 12 by 12 descriptor with 3 by 3 subwindowing. This is expected behavior because the 16 by 16 optimal size was selected after much prior experimentation and is the one used in standard SIFT implementations.

4 Conclusion

In this lab assignment, the invariance of the features provided by SIFT descriptor [1] is studied, analysed, and tested with the dataset created in the first section of this report. This dataset consists of 3 sequences (projective transformation, zoom, and rotation) with distinct number of images. Subsequently, figures that show the percentage of the correct matches between the reference image and the transformed images in each sequence are generated and conclusions are drawn. The scale and rotation invariance of SIFT is observed, as also the partial invariance to 3D viewpoint changes. Moreover, a modified version of SIFT is implemented using the code provided by [3] and further comparisons are made with a different size of the descriptor using the same previous dataset.

References

[1] David G. Lowe. Image Features from Scale-Invariant Keypoints . 4^{th} January 5, 2004.

[2] http://www.cs.ubc.ca/lowe/keypoints/~lowe/keypoints/.

[3] http://www.vlfeat.org/.