ISSN:2231-0711
www.ijcset.net

Modified Sift Algorithm for Appearance Based Recognition of American Sign Language

Jaspreet Kaur ,Navjot Kaur

*Electronics and Communication
Engineering Department
I.E.T. Bhaddal, Ropar,
Punjab,India.*

Abstract: **The concern of the paper is to investigate the application of the Scale-Invariant Feature Transform (SIFT) to the problem of hand gesture recognition by using MATLAB. The algorithm uses modified SIFT approach to match key-points between the query image and the original database of Bare Hand images taken. The extracted features are highly distinctive as they are shift, scale and rotation invariant. They are also partially invariant to illumination and affine transformations. All these properties make them very suitable to the problem at hand. Performance improvement for SIFT also has been proposed. Experimental results show the efficient performance of the developed algorithm in terms of recognizing all the images provided in the training set.**

Keywords: SIFT, ASL Recognition, ASL using MATLAB, Image Processing

1. INTRODUCTION

Sign language can be considered as a collection of gestures, movements, posters, and facial expressions corresponding to letters and words in natural languages. The sign language is the fundamental communication method between the people who suffer from hearing defects. In order for an ordinary person to communicate with deaf people, a translator is usually needed to convert the sign language into natural language and vice versa. The aim of sign language alphabets recognition is to provide an easy, efficient and accurate mechanism. With the help of computerized digital image processing and MATLAB, the system can interpret ASL alphabets.

It attempts to process static images of the subject considered, and then matches them to a statistical database of pre-processed images to ultimately recognize the specific set of signed letters. Since the approach taken in this analysis is vision-based, the amount of processing is minimized as compared to other approaches and hence projects itself as a viable technique to be implemented in real time systems. I intend to describe the approach to demonstrate the results thus derived, where several words are distinguished and recognized with a fairly high degree of reliability.

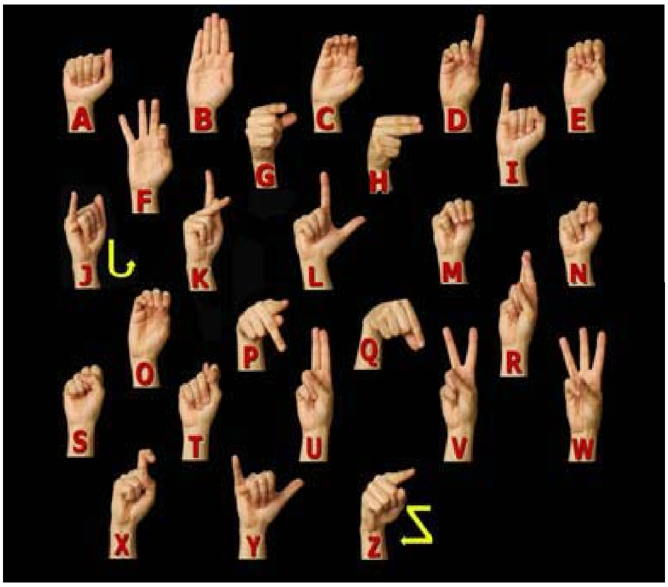


Figure 1: ASL Finger Spelling Alphabet

2. BACKGROUND

Comparing images in order to establish a degree of similarity is an important computer vision problem and has application in various domains such as robot localization, content-based medical image retrieval, and image registration.

Comparing images remains a challenging task because of issues such as variation in illumination conditions, partial occlusion of objects, differences in image orientation etc. Global image characteristics such as colour histograms, responses to filter banks etc. are usually not effective in solving real-life image matching problems.

Researchers have recently turned their attention to local features in an image, which are invariant to common image transformations and variations. Usually two broad steps are found in any local feature-based image-matching scheme. The first step involves detecting features (also referred to as key points or interest points) in an image in a repeatable way.

Repeatability is important in this step because robust matching cannot be performed if the detected locations of key points on an object vary from image to image. The second step involves computing descriptors for each detected interest point. These descriptors are useful to distinguish between two key points.

The goal is to design a highly distinctive descriptor for each interest point to facilitate meaningful matches, while simultaneously ensuring that a given interest point will have the same descriptor regardless of the hand position, the lighting in the environment, etc. Thus both detection and description steps rely on invariance of various properties for effective image matching.

Image matching techniques based on local features are not new in the computer vision field. Van Gool [5] introduced the generalized colour moments to describe the shape and the intensities of different colour channels in a local region. Sven Siggelkow [2] used feature histograms for content-based image retrieval. These methods have achieved relative success with 2D object extraction and image matching. Mikolajczyk and Schmid [3] used the differential descriptors to approximate a point neighbourhood for image matching and retrieval. Schaffalitzky and Zisserman [4] used Euclidean distance between orthogonal complex filters to provide a lower bound on the Squared Sum Differences (SSD) between corresponding image patches. David Lowe proposed [1] Scale Invariant Feature Transforms (SIFT), which are robustly resilient to several common image transforms. Mikolajczyk and Schmid reported an experimental evaluation of several different descriptors where they found that the SIFT descriptors obtain the best matching results.

3. Modified SIFT

In this section, we describe the SIFT algorithm [1, 6] in more detail. We also state the modifications that were made to increase simplicity. Since, we have a small database of bare hands, these simplifications were quite reasonable and didn't affect the accuracy of the algorithm.

SIFT is an approach for detecting and extracting local feature descriptors that are reasonably invariant to changes in illumination, image noise, rotation, scaling, and small changes in viewpoint.

A complete description of SIFT can be found in [1].An overview of the algorithm is presented here. The algorithm has the major stages as mentioned below:

- Scale-space extrema detection: The first stage searches over scale space using a Difference of Gaussian function to identify potential interest points.
- Key point localization: The location and scale of each candidate point is determined and key points are selected based on measures of stability.
- Orientation assignment: One or more orientations are assigned to each key point based on local image gradients.
- Key point descriptor: A descriptor is generated for each keypoint from local image gradients information at the scale found in the second stage.

Each one of the above-mentioned stages is elaborated further in the following sections.

A. FINDING KEYPOINTS

The SIFT feature algorithm is based upon finding locations (called key points) within the scale space of an image which can be reliably extracted. The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation. Key points are identified as local maxima or minima of the DoG images across scales.

Each pixel in a DoG image is compared to its 8 neighbours at the same scale, and the 9 corresponding neighbours at neighbouring scales. If the pixel is a local maximum or minimum, it is selected as a candidate key point.

We have a small image database, so we don't need a large number of key points for each image. Also, the difference in scale between large and small bare hands is not so big.

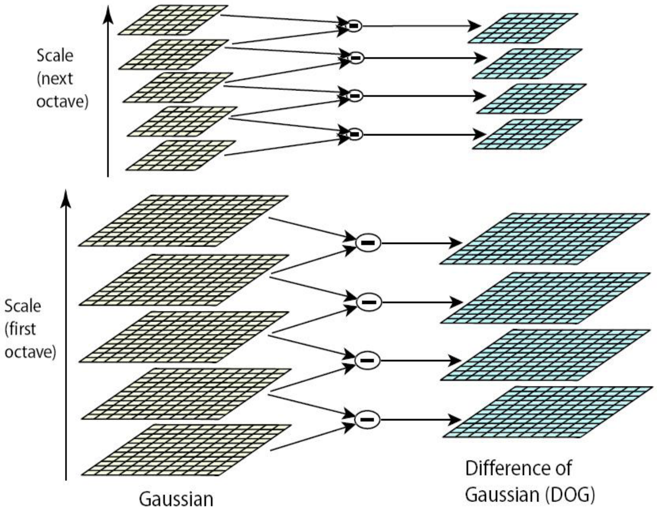


Figure2: Gaussian & DoG pyramids (Source: Reference 1)

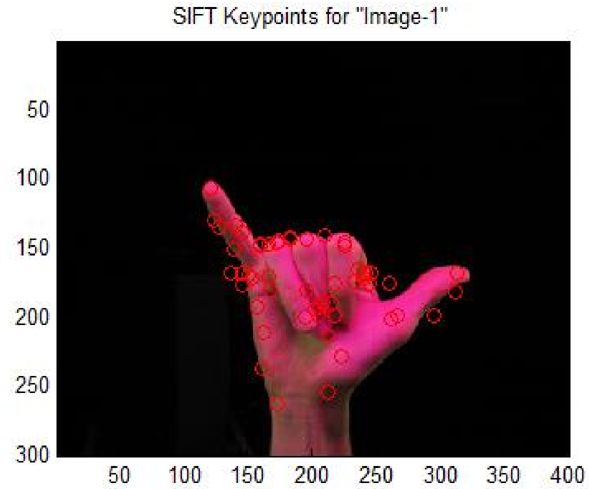


Figure3: Detected key points for Image representing "Y" Character

B. KEYPOINT LOCALIZATION

In this step the key points are filtered so that only stable and more localized key points are retained. First a 3D quadratic function is fitted to the local sample points to determine the location of the maximum. If it is found that the extremum lies closer to a different sample point, the sample point is changed and the interpolation performed instead about that point. The function value at the extremum is used for rejecting unstable extrema with low contrast.The DoG operator has a strong response along edges present in an image, which give rise to unstable key points. A poorly defined peak in the DoG function will have a large principal curvature across the edge but a small principal curvature in the perpendicular direction.

C. ORIENTATION ASSIGNMENT

In order for the feature descriptors to be rotation invariant, an orientation is assigned to each key point and all subsequent operations are done relative to the orientation of the key point. This allows for matching even if the query image is rotated by any angle. In order to simplify the algorithm, we tried to skip this part and assume no orientation for all key points. When tested , it gave wrong results with nearly all the images where the bare hand image is rotated with an angle of 15º to 20º or more. We realized that this step can't be eliminated. In this algorithm, the orientation is in the range [-PI, PI] radians.

D. KEYPOINT DESCRIPTORS

First the image gradient magnitudes and orientations are calculated around the key point, using the scale of the key point to select the level of Gaussian blur for the image. The coordinates of the descriptor and the gradient orientations are rotated relative to the key point orientation. Note here that after the grid around the key point is rotated, we need to interpolate the Gaussian blurred image around the key point at non-integer pixel values. We found that the 2D interpolation in MATLAB takes much time.

So, for simplicity, we always approximate the grid around the key point after rotation to the next integer value. By experiment, we realized that, this operation increased the speed much and still had minor effect on the accuracy of the whole algorithm. The gradient magnitude is weighted by a gaussian weighting function with σ , equal to one half of the descriptor window width to give less credit to gradients far from center of descriptor. Then, these magnitude samples are accumulated into an orientation histogram summarizing the content over 4x4 subregion.

Fig. 4 describes the whole operation. Trilinear interpolation is used to distribute the value of each gradient sample into adjacent bins. The descriptor is formed from a vector containing the values of all the orientation histogram entries. The algorithm uses 4x4 array of histograms with 8orientation bins in each resulting in a feature vector of 128 elements. The feature vector is then normalized to unit length to reduce the effect of illumination change.

The values in unit length vector are thresholded to 0.2 and then renormalized to unit length. This is done to take care of the effect of nonlinear illumination changes.

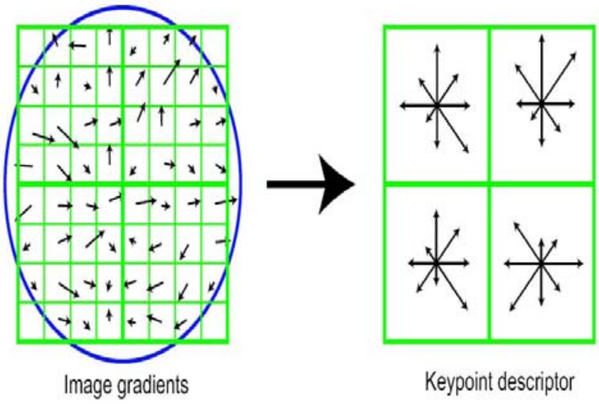


Figure 4: 2x2 descriptor array computed from 8x8 samples (Source: Reference 1)

E. SIMPLIFICATIONS TO SIFT ALGORITHM

The distance of one feature point in first image and all feature points in second image must be calculated when SIFT algorithm is used to match image, every feature point is 128-dimensional data, the complexity of the calculation can well be imagined.

A changed Comparability measurement method is introduced to improve SIFT algorithm efficiency. first, Euclidean distance is replaced by dot product of unit vector as it is less computational.; then, Part characteristics of 128-dimensional feature point take part in the calculation gradually. SIFT algorithm time reduced.

Euclidean Distance is distance between the end points of the two vectors. Euclidean distance is a bad idea because Euclidean distance is large for vectors of different lengths. This measure suffers from a drawback: two images with very similar content can have a significant vector difference simply because one is much longer than the other.

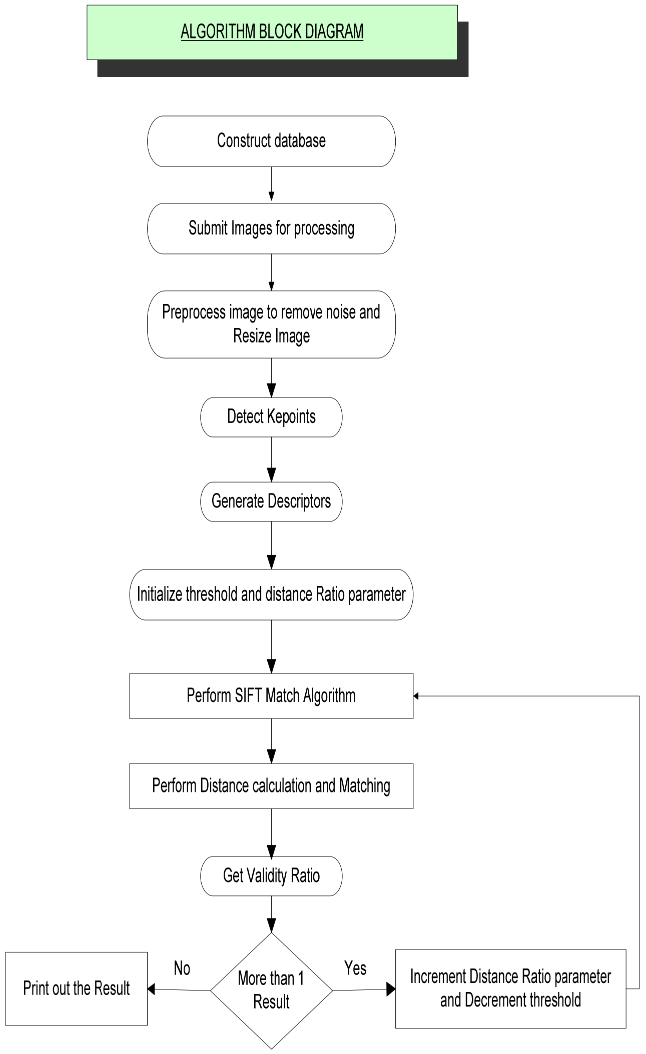
Thus the relative distributions may be identical in the two images, but the absolute term frequencies of one may be far larger. So the key idea is to rank images according to angle with query images. To compensate for the effect of length, the standard way of quantifying the similarity between two images $d1$ and $d2$ is to compute the *cosine similarity* of their vector representations V($d1$) and V($d2$)

$$\text{sim}\,(d1, d2) = V\,(d1).\ V\,(d2) \,/\, |V\,(d1)\;||V\,(d2)|$$

where the numerator represents the dot product (also known as the inner product) of the vectors V (d1) and V (d2), while the denominator is the product of their Euclidean lengths.

4. KEYPOINT MATCHING USING UNIT VECTORS

1. Match (image1, image2). This function reads two images, finds their SIFT [1] [6] features. A match is accepted only if its distance is less than dist Ratio times the distance to the second closest match. It returns the number of matches displayed.


ALGORITHM BLOCK DIAGRAM
Construct database
Submit Images for processing
Preprocess image to remove noise and Resize Image
Detect Kepoints
Generate Descriptors
Initialize threshold and distance Ratio parameter
Perform SIFT Match Algorithm
Perform Distance calculation and Matching
Get Validity Ratio
More than 1 Result
No
Yes
Print out the Result
Increment Distance Ratio parameter and Decrement threshold

2. Find SIFT (Scale Invariant Fourier Transform) Key points for each image. For finding the SIFT Key points specify what are its locations and descriptors.
3. It is easier to compute dot products between unit vectors rather than Euclidean distances. Note that the ratio of angles acos of dot products of unit vectors is a close approximation to the ratio of Euclidean distances for small angles.
4. Assume some distance ratio for example suppose distance ratio=0.5 it means that it only keep matches in which the ratio of vector angles from the nearest to second nearest neighbour is less than distance Ratio.
5. Now for each descriptor in the first image, select its match to second image.
6. Compute matrix transpose, Computes vector of dot products, Take inverse cosine and sort results. Check if nearest neighbour has angle less than dist Ratio times second.
7. Then create a new image showing the two images side by side.

Now apply these steps in our previous image from which SIFT features are extracted.

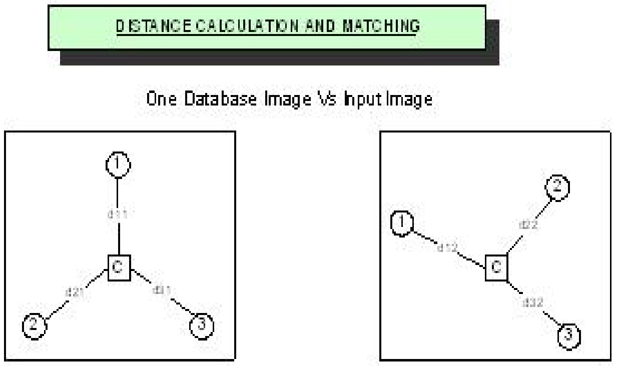

DISTANCE CALCULATION AND MATCHING
One Database Image Vs Input Image
1
2
3
C
d11
d21
d31
1
2
3
C
d12
d22
d32

Figure 5: One database image versus input image

$$d_{T1} = \sum_{i=1}^{m} di1 \text{ ------ (1)}$$

$$d_{T2} = \sum_{i=1}^{m} di2 \text{ ------ } \qquad (2)$$

Ratios1 = $[d_{11}/d_{T1,}\, d_{21}/d_{T1,}\, d_{31}/d_{T1}]$ ------ (3)

Ratios2 = $[d_{12}/d_{T2,}\, d_{22}/d_{T2,}\, d_{32}/d_{T2}]$ ------ (4)

Distance = abs [Ratio1 – Ratio2] < matching Threshold --- (5)

Valid Points = sum (Distance) ------ (6)

Using this algorithm we read image and calculate key- points, descriptors and locations by applying threshold. Descriptors given as P-by-128 matrix where p is number of key-points and each row gives an invariant descriptor for one of the p key-points. The descriptor is a vector of 128 values normalized to unit length. Locations are P-by-4 matrix, in which each row has the 4 values for a key-point location (row, column, scale, orientation). The orientation is in the range [-PI, PI] radians.

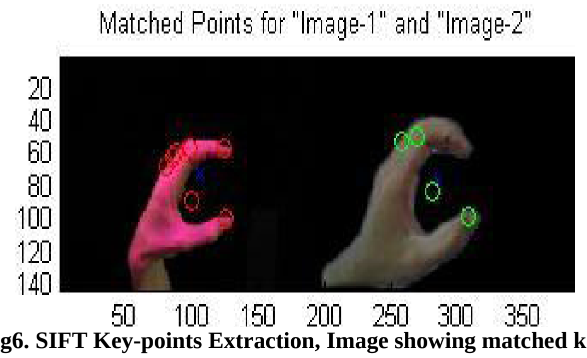


Fig6. SIFT Key-points Extraction, Image showing matched key-points between input image and database image.

The problem now is how we can identify a 'No Match'. For this, we saw that the 'No Match' query images are in many cases confused with the database images that have a large number of feature vectors in the feature vectors database. We decided to compare the highest vote (corresponding to the right image) and the second highest vote (corresponding to the most conflicting image). If the difference between them is larger than a threshold, then there is a match and this match corresponds to the highest vote. If the difference is smaller than a threshold, then we declare a 'No Match'. The values of THRESHOLD were chosen by experiment on training set images either with match or no match.

5. Implementation

The approach described above has been implemented using MATLAB. The implementation has two aspects: training and inference. During the training phase locally invariant features (key points, orientation, scales and descriptors) from all training images are retrieved using the SIFT algorithm .During inference, the objective is to recognize a test image. A set of local invariant features are retrieved for the test image during the inference phase and compared against the training feature-set using the metric explained in section 4. The title of the closest match is returned as the final output.

6. Result And Analysis

In order to prove the performance of our proposed system, we predefined the number of gestures from B, C, H, I, L, O, Y and create a hand gesture database. Matching is performed between images by unit vectors. The matching is accomplished for proposed method and the result shows that it produces 98% accuracy. In Figure 7, we can easily identify Database Images 1, 3, 7 have more number of key-points matched with input image key-points .So Distance Ratio parameter and threshold are adjusted.

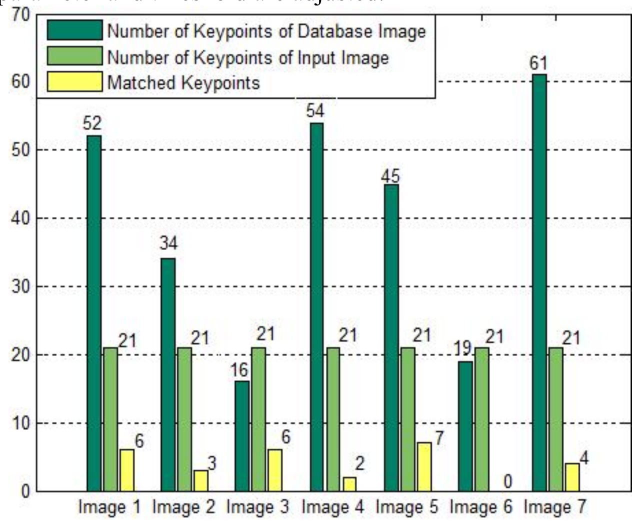


Figure 7: comparison of key-points on given input with database for a single input image for first cycle

In Figure 8, we compare Database Images 1, 3, 7 with input image key points. So Database Image 3 is closest match with input image.

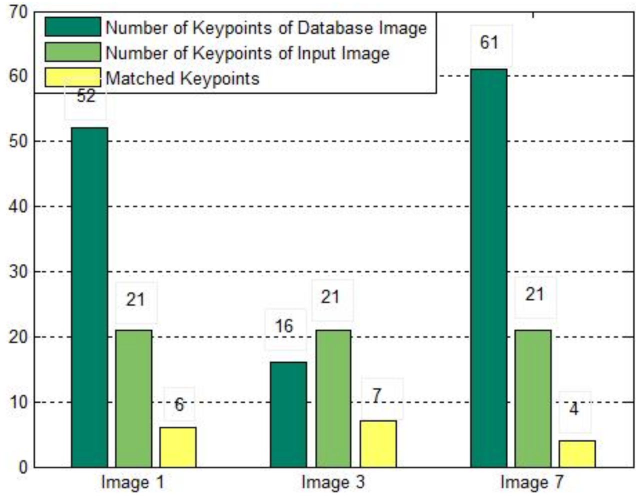


Figure 8: comparison of key-points on given input with database for a single input image after resetting threshold value and distance ratio value

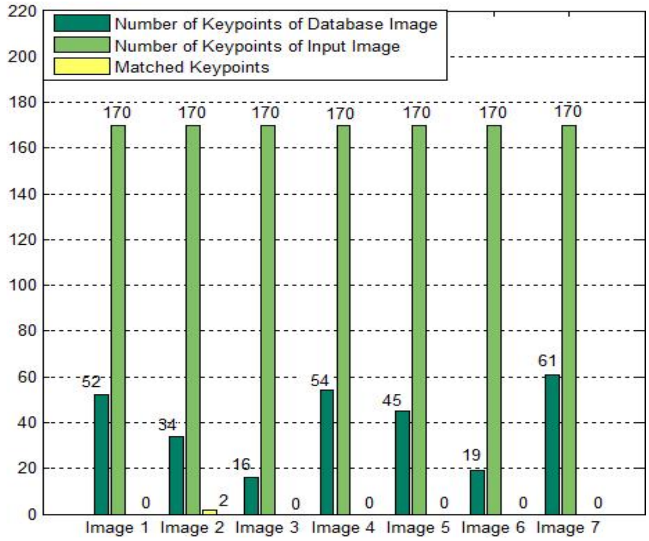


Figure 9: comparison of key-points on given input with database for a single input image(No Match Case)

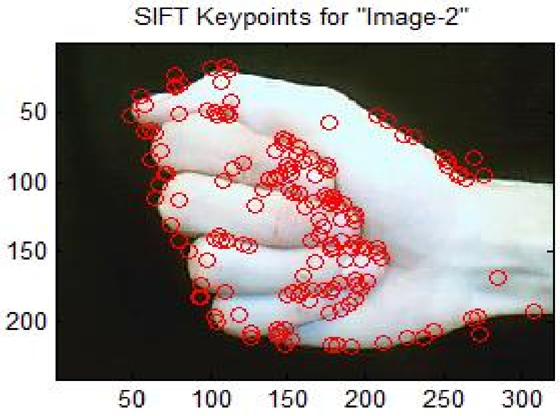


Figure10: Example of a " no match" Image not in training set for figure3

Gesture Name	Testing Number	Success Number	Correct Rate
B	150	149	99.3
C	150	148	98.7
H	150	148	98.7
I	150	149	99.3
L	150	148	98.7
O	150	148	98.7
Y	150	149	99.3

Table1. The results of classifier for the training set and testing set

7. CONCLUSIONS

The Algorithm is based mainly on using SIFT features to match the image to respective sign by hand gesture. Some modifications were made to increase the simplicity of the SIFT algorithm. Applying the algorithm on the training set, we found that it was always able to identify the right sign by hand gesture or to declare 'No Match' in case of no match condition. The algorithm was highly robust to scale difference, rotation by any angle and reflection from the test image. SIFT is a state-of-the-art algorithm for extracting locally invariant features and it gave me an opportunity to understand several aspects of application in image recognition. I believe this effort resulted in a robust image recognition implementation, which should perform quite well with the final test images. In future I would like to work on improving the performance of the SIFT for Global Features. The local invariant features of SIFT can be augmented by computing global features of an image.

REFERENCES

[1] David G. Lowe. Distinctive Image Features from Scale-Invariant Key points. International Journal of Computer Vision, 60, 2 (2004), pp.91-110.

[2] Sven Siggelkow. Feature Histograms for Content-Based Image Retrieval. PhD Thesis, Albert-Ludwigs-University Frieiburg, December 2002.

[3] Mikolajczyk, K.,Schmid, C.: An Affine Invariant Interest Point Detector. In: ECCV, (2002) 128-142

[4] Schaffalitzky, F., Zisserman, A.: Multi-view Matching for Unordered Image Sets, or "How Do I Organize My Holiday Snaps?" In: ECCV, (2002) 414-431

[5] Van Gool, T. Moons, and D. Ungureanu. Affine photometric invariants for planar intensity patterns. In ECCV, pp. 642-651, 1996.

[6] D. Lowe, "Object Recognition from Local Scale-Invariant Features," *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999.

[7] M. Atallah, Y. Genin, and W. Szpankowski, "Pattern matching Image compression: Algorithmic and empirical results," IEEE Trans. Pattern Anal. Machine Intell., vol. 21, pp. 614–627,1999.