COMPUTER LAB1: SIFT

Ethelbert Uzodinma (S3886026)
Viljami Linna (S4169921)

ABSTRACT

SIFT is an acronym for Scale-Invariant Feature Transform. It is a feature extraction technique used in Computer Vision to detect and describe local features from Images.The algorithm is widely applied in real world in applications which include but not limited to : face recognition, Image recognition, robotic navigation and mapping, object and video tracking etc. In this report various images are matched with their location in in a clutter scene, containing other objects with various orientation.

Index Terms— SIFT, object recognition, clutter, template matching, feature extraction, key points, computer vision

1. INTRODUCTION

Feature mapping in object recognition has remained a puzzle yet to be solved completely in the field of computer vision. Most especially in dealing with Images from a clutter scene that contains objects of various scales and orientations, at this point most simple corner detectors like the: Harris detectors may not be very reliable or must be used with SIFT for more reliability.

Given a particular object in an image, regions of interest can be extracted to give the objects description of its features using the SIFT algorithm, this same object features in the training image can be used to identify the object when trying to locate it in a test image (cluttered scene) containing other images.
For reliable recognition this features extracted from the training image should be detectable and invariant even in higher resolution of the image or in a noisy image.

SIFT can robustly identify objects even among cluttered or partially occluded scenes, because the SIFT algorithms descriptor is invariant to changes in orientation, scale, illumination and affine distortion.

The Image Data provided for this experiment is compressed in a zipped folder named: **siftprakt.zip**. Using this Image data we are to test how good the SIFT method is in extracting features from images more robustly.

2. METHOD

For the practical we first download the file "siftprakt.zip" into the work directory of MatLab.
Then type:

```
cd siftDemoV4
```

We compute the SIFT features by typing
`[I,keys,loc] = sift("example.pgm")`

We made sure that the filename must specify a gray scale image.Since its a requirement to work with SIFT algorithm.
The `I` variable contains the Image, the `keys` contains the key point features, the `loc` contains the location and orientation of the key points.
To display the key points:
`showkeys(I,loc)`
is entered in the MATLab command window. This represent the "points of interest" on the image that must be displayed no matter how the image is distorted. They contain Information about the image
Also we use the function:
`match('xxxx.png','xxxx.png')`

to match the patterns in image to the corresponding pattern in the clutter image.The above commands is used in our experiment using the MATLab to test the reliability of the SIFT commands.

2.1. Exercise1

In this exercise we computed SIFT features of "`scene.pgm`" and compared our results with the results in the paper [1]. The size of the image was `384 x 512` pixels as opposed to the image in the paper which was a `512x512 pixel` pixel (see Figure 1).

The number of key points can be got by using the `sift()` above in this case:
`[I,keys,loc] =sift("scene.pgm")`
The `keys` variable contains `1021 x 128` matrix in which `1021` represents the number of key points which is similar as the results in the paper which is gives on the order of `1000` key points[1].

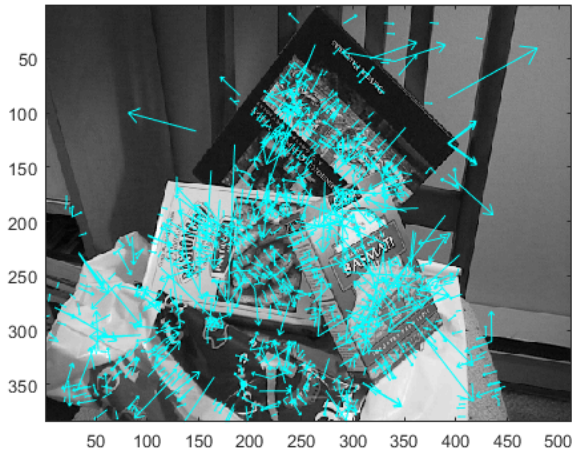

Fig. 1. A figure showing the various key points in an image clutter

The other dimension of the keypoint matrix represents the number of features of a keypoint, which was `128`. Instead, in the paper `160` features were used calculating one keypoint. Different variations of the SIFT method can be executed modifying several parameters in it. This can also have an affect to the number of features per keypoint. However, we were not able to take a close look at the source code of the provided algorithm to find the exact reason behind the difference.

Image and its keypoints are represented in Figure 1. The base of the blue arrow represents the position of the key point. The length of the arrow depends on the area where the features of the key point are collected.

As displayed in Figure 1 most of the key points are located in the relevant areas of the image where most of the unique features occur. Only a few key points are located in the blank areas of the image. Hence the quality of the key points is good and most of them can be utilized for object recognition.

We tested key point detection reliability applying `"scene.pgm"` and `"book.pgm"` to the `match()` function which extracts the key points of both images and the find matches between the points. We found `1021` key points from the first, and `882` key points from the second image. There were `98` matches of key points between the images. The result of `match()` function is displayed in Figure 2. We repeat same for `"scene.pgm"` and `"basmati.pgm"`. Result of the `match()` function with these parameters is represented in Figure 3.

According to the results, the matching algorithm found a match of approximately 10% of the key points. The amount of the mismatches were 2-3% of all matches. Calculation results are represented in Figure 4. The number of correct matches was observed on the images side-by-side and seen to be less than the total.

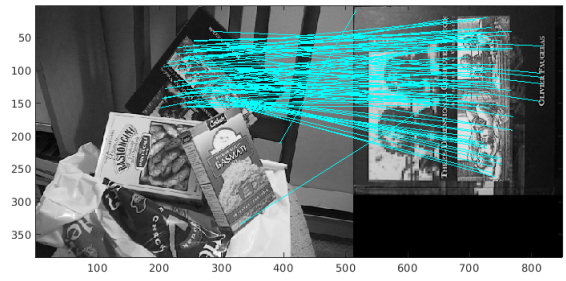

Fig. 2. Matches between 'scene.pgm' and 'book.pgm.'

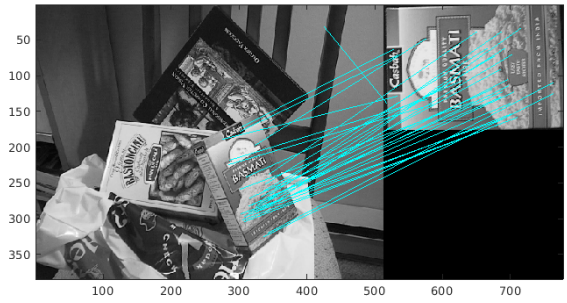

Fig. 3. Matches between 'scene.pgm' and 'basmati.pgm.'

The mismatches can be said to be reasonable because the algorithm observed the similar patterns on both images i.e global features such as edges or colours which seems important and are similar though unrelated to the region of interest and highlights it.

Picture:	scene.pgm	book.pgm	basmati.pgm
Keypoints:	1021	882	579
Matches	98	98	34
Matches/keypoints (%)	9,60	11,11	5,87
Mismatches		2	1
Mismatches/matches (%)		2,04	2,94

Fig. 4. Matches and key points between compared images.

Also when we compare the two street images "`street.png and streetlarge.png`" as shown in the Figure 5. We can observe that `streetlarge.png` with a resolution of `720 x 1080` has a total of `5316` key points while the smaller `street.png` has `1452` key points.

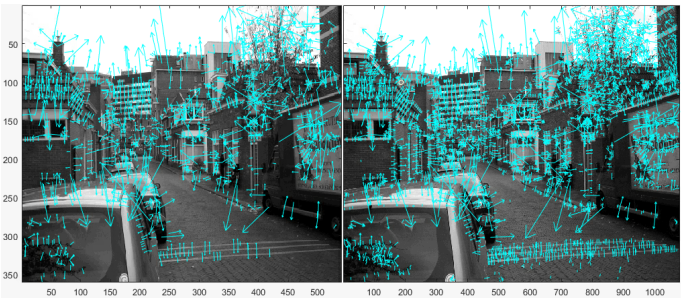

Fig. 5. Comparing number of key points 'street.png' and 'streetlarge.png'

We then compare the performance of SIFT matching on higher resolution images. From the results in Figure 11 and Figure 12, it can be seen that higher resolution images gives more matches but does not improve the number of matches as more unrelated features are matched. Results of comparisons are displayed in Figure 6 and Figure 7.

Matched Image	Resolution (pixel)	No of key points	No of Matches	% of Match w.r.t total key points	No of correct matches
detail1.png	342 x 432	1592	25	1.6	24
detail2.png	126 x 138	46	6	13.0	6
detail3.png	300 x 204	588	37	6.3	37
detail4.png	216 x 255	368	20	5.4	18
detail5.png	87 x 102	126	6	4.8	6

Fig. 6. Keypoints and matches 'street.png.'

Matched Image	Resolution (pixel)	No of key points	No of Matches	% of Match w.r.t total key points	No of correct matches
detail1.png	342 x 432	1592	82	5.2	79
detail2.png	126 x 138	46	17	36.9	15
detail3.png	300 x 204	588	110	18.7	86
detail4.png	216 x 255	368	73	20	59
detail5.png	87 x 102	126	17	13.5	13

Fig. 7. Keypoints and matches 'streetlarge.png.'

2.2. Exercise2

In this exercise, we set a threshold for a claim that an image match with another image. We applied five detail images which all occur in `'street.png'` to the `match()` function with `'scene.png'` and `'street.png.'` For each comparison, we determined a total number of keypoints in detail image and number of matches between compared images. The results is displayed in Figure 8.

Based on the results we set the threshold for the claim "there is a match between the images" to 1% thus agree images matching only if match found between at least 1% of the keypoints. The result cannot be considered reliable since we had only ten data points.

	Keypoints	Matches with scene.png	Matches with street.png	Matches with scene.png(%)	Matches with street.png(%)
detail1	1592	1	25	0,06	1,57
detail2	46	0	26	0,00	56,52
detail3	588	0	37	0,00	6,29
detail4	368	2	20	0,54	5,43
detail5	126	0	6	0,00	4,76

Fig. 8. Amount of matches when comparing details to 'scene.pgm' and 'street.png.'

2.3. Exercise3

We applied `scene.pgm` with rotated `book.pgm` to the `match()` function. We plot amount of matches against rotation angle represented in Figure 9. According to the results, SIFT key point detection and key point descriptors are rotation invariant methods.

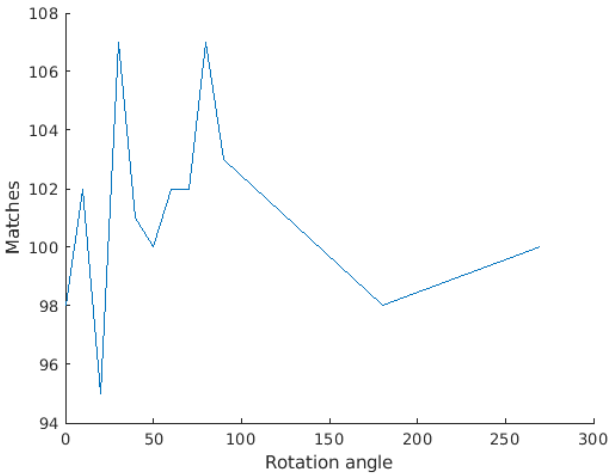

Fig. 9. Matches between 'scene.pgm' and 'book.pgm' with different rotation angles.

2.4. Exercise4

In this exercise we sheared image `'book.pgm'` with different factors and then tried to find common key points with image `'scene.pgm'` using `match()` function. We plot matches against shearing shown in Figure 10. According to the results, shearing has an effect of key point detection and descriptors. The number of matches decreased when shearing factor increased.

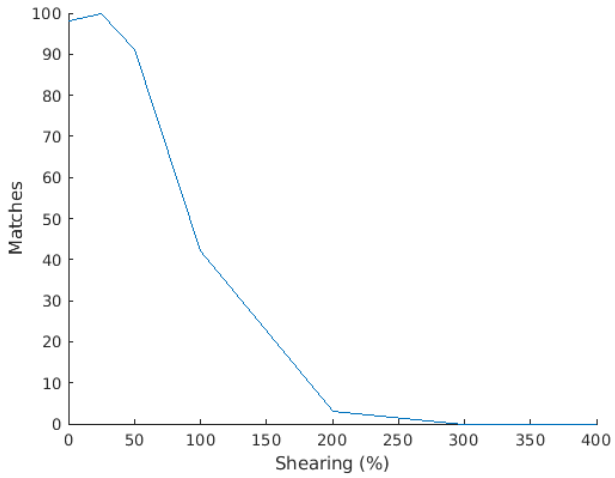

Fig. 10. Matches between 'scene.pgm' and 'book.pgm' with different shearing factor.

3. RESULTS

After matching the `street.png` and the `streetlarge.png` images with the `details*.png` images. below the results are displayed in the tables.

Matched Image	Resolution (pixel)	No of key points	No of Matches	% of Match w.r.t total key points	No of correct matches
detail1.png	342 x 432	1592	25	1.6	24
detail2.png	126 x 138	46	6	13.0	6
detail3.png	300 x 204	588	37	6.3	37
detail4.png	216 x 255	368	20	5.4	18
detail5.png	87 x 102	126	6	4.8	6

Fig. 11. Results of matching street.png with the details*.png

Matched Image	Resolution (pixel)	No of key points	No of Matches	% of Match w.r.t total key points	No of correct matches
detail1.png	342 x 432	1592	82	5.2	79
detail2.png	126 x 138	46	17	36.9	15
detail3.png	300 x 204	588	110	18.7	86
detail4.png	216 x 255	368	73	20	59
detail5.png	87 x 102	126	17	13.5	13

Fig. 12. Results of matching streetlarge.png with the details*.png

4. REFERENCES

[1] David G. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157 vol.2, 1999.