CHAPTER 4
COPY-MOVE FORGERY DETECTION BASED ON HARRIS INTEREST POINTS AND SIFT DESCRIPTORS

4.1. Introduction

The main task of matching features that are defined by interest points is to achieve invariance to the mentioned changes. In this context, the term feature descriptor is often used, denoting the data structure that is compared in order to calculate the similarity between two feature points. Various methods have been proposed and are available in literature for this purpose. In [BA00], an approach is proposed to calculate the moment descriptor using a rotationally symmetric Gaussian window function. In another approach [SM97], local jets according to [KD87] are used to compute multiscaled differential grayvalue invariants. Tuytelaars et. at. [TV04] proposed two types of affinity invariant regions, one based on the combination of interest points and edges, and the other based on image intensities. Mikolajczyk et at. [MS05] evaluated the performance of five types of local descriptors: SIFT, Steerable Filters [FA91], differential invariants [KD87], complex filters [SZ02], and Moment Invariants [GMU96] and the local affine region detectors are surveyed in [MTS05]. According to their experiment, except for light changes, the SIFT descriptor outperforms the other descriptors.

Murphy-Chutorian et. at. [MCT05], presented a method which uses the Gabor wavelet transformation around Shi-Tomasi interest points in order to calculate a feature descriptor for an object recognition system with a database of 50 objects. K-means clustering is used to reduce the number of features stored in the database. A completely different approach was presented by Lepetit, et. at. [LPF04] for point matching. In their approach instead of calculating a descriptor analytically to achieve invariance, robustness to scaling, rotation, and skew is achieved in a brute-force manner. Each image patch around a point feature is represented by a set of synthetically generated different views of the same patch, intended to cover all possible views. In order to speedup matching, PCA is applied to all view sets. Point matching is performed by calculating the nearest neighbor in
the Eigen space for a given image patch. In this chapter, a new approach, which combines the Harris corner detector with the SIFT descriptor to detect copy-move forgery is introduced.

4.2. HARRIS-SIFT Forgery Detection System

Good copy-move forgery detection should be robust to some type of transformations. One of the main strengths of SIFT features are their scale invariance. However, the scale-space analysis required for the calculation of the SIFT feature point positions is too slow for visual related applications. In our approach we focus on detection of copy-move forgery which is robust to some types of manipulation based on Harris corner detector and SIFT descriptors. A simple schematization of the whole system is shown Figure 4.1. The first phase consists of interest point detection using Harris corner detector, SIFT features extraction in the second phase and keypoint matching is done in the third phase using kd-tree.

Figure 4.1: Overview of HARRIS-SIFT Forgery Detection System
4.2.1. Keypoints Detection

As already stated, the SIFT descriptor is a very robust and reliable representation for the local neighborhood of an image point. However, the scale-space analysis required for the calculation of the SIFT feature point positions is too slow for visual serving applications. One of the main strengths of the SIFT features are their scale-invariance. This is achieved by analyzing and processing the images at different scales. For this, a combination of Gaussian smoothing and a resize operation is used. Between two so-called octaves, the image size is halved, i.e. resized to half width and half height. The different scales within an octave are produced by applying a Gaussian smoothing operator, and the variance of the Gaussian kernel is chosen in a way that the last scale of one octave and the first scale of the next octave correspond to each other. Since the scale space analysis performed by the SIFT features for calculating the feature point positions is the by far most time-consuming part, the idea is to replace this step by a faster method, namely an appropriate corner detector. Different Interest points detector have been proposed and used based on the field of applications. The fast, robust and rotation invariant, Harris detector is widely used in many computer vision applications which uses the autocorrelation function to determine locations where the change of signal in one or two directions. A matrix related to the auto-correlation function is computed:

\[
C(x, \sigma_I, \sigma_D) = \sigma_I^2 G(x, \sigma_I) \ast \begin{pmatrix}
L_x^2(x, \sigma_D) & L_x L_y(x, \sigma_D) \\
L_x L_y(x, \sigma_D) & L_y^2(x, \sigma_D)
\end{pmatrix}
\]  \hspace{1cm} (1)

where \( \sigma_D \) is the derivation scale, \( \sigma_I \) is the integration scale, \( G \) is the Gaussian and \( L \) is the image smoothed by a Gaussian kernel. This matrix has two Eigen values that are the principal curvatures of the auto-correlation function. When the two eigenvectors are very small then there is no structure exists. If one is large and another one is small, there is an edge like structure. If both of them are very large and distinct, there is a corner like structure. Edges and interest points can be computed based on:
Edges are computed based on equation (2), where $\alpha$ is the coefficient of the Harris function and $T_E$ is the threshold of the Harris function ($T_E < 0$). The edge detection is carried out at the first scale. Interest points can be detected by using eq. (3), $T_C$ is the threshold for interest points ($T_C > 0$).

**4.2.2. Feature Extraction**

The Feature extraction is the main for any system which requires matching. The extracted features should be well separated in the feature space to produce effective discrimination between images. Once these keypoints are detected using Harris detector, and canonical orientations are assigned, SIFT descriptors are computed at their locations in both image plane and scale-space. Each feature descriptor consists of a histogram $f$ of 128 elements, obtained from a 16 x 16 pixel area around the corresponding keypoint.

**4.2.3. Key Point Matching**

Image matching plays an important role in many fields such as computer vision, robotics, image recognition etc. There are several matching algorithm which are represented in literature. The matching algorithm could be broadly classified as Area Based Matching (ABM) and Feature Based Matching (FBM). The Area Based Matching requires initial values for the unknown parameter for matching. The feature based matching algorithm determines the correspondence between image features and it does not require initial estimates. Unlike the area based matching, the feature based matching only the selected points with certain
features are to be matched. Combining Harris detector and SIFT descriptors the unique features of the image are extracted which are distinct in nature. Moreover, the selections as well as the selected position are invariant to geometric transformation.

a) Kd-tree

There are many problems in image processing which requires fast analyses and fast search in multidimensional data. A most popular technique Kd-tree is one such solution for fast analyses and search in multidimensional data. The main advantage of Kd-tree is that easy to build and has a simple algorithm for closest points and ranged search. In the proposed system, to identify the duplication region the KD-tree algorithm is used for key points matching. In most of the copy-move forgery detection algorithms, lexicographic sorting are used, which is said to be too sensitive to the transformations and yields a lower false positive rate compared to KD-Tree which produces reliable results and a lower false negative rates. The KD tree pre-processes data into a data structure allowing us to make efficient range queries. The region matching in Harris-SIFT Forgery system are matched based on threshold based matching technique. In Harris-SIFT Forgery Detection System the distance between the regions are compared to a given distance threshold $D_t$. 

Please purchase PDF Split-Merge on www.verypdf.com to remove this watermark.
Input image  
//Keypoint extraction  
Convert color image to grayscale image  
a[i] = Keypoints extraction from the image using HARRIS detector  
n = no. of keypoints extracted based on HARRIS threshold  
// Descriptor Extraction  
For i=1 to n  
Vectorarray [point] ← Extract 128 dimension vector for a[i] using SIFT  
End  
// Constructing Kd Tree  
CONSTRUCTKDTREE(Vectorarray[point], depth)  
If Vectorarray[point] contains only one point  
then return a leaf storing this point  
Else if depth is even  
then Spilt Vectorarray[point] into two subsets with a vertical line $l$  
through the median x-coordinate of the points in Vectorarray[point]  
.  
Let Vectorarray[point]_1 be the set of points to the left of $l$ or on $l$, and  
let Vectorarray[point]_2 be the set of points to the right of $l$.  
Else Spilt Vectorarray[image] into two subsets with a horizontal line $l$  
through the median y-coordinate of the points in Vectorarray[point]  
Let Vectorarray[image]_1 be the set of points below $l$ or on $l$, and  
Let Vectorarray[image]_2 be the set of points above $l$.  
Vectorarray[point]_1 ← CONSTRUCTKDTREE(Vectorarray[image]_1, depth+1)  
Vectorarray[point]_2 ← CONSTRUCTKDTREE(Vectorarray[image]_2, depth+1)  
Create a node V storing $l$, making Vectorarray[point]_1 the left child of V, and make Vectorarray[point]_2 the right child of V.  
// Matching  
$T_h = Kd$ threshold  
If one node $\approx$ another node (based on distance threshold $D_t$)  
then “Copied Region”  
else  
“Not Copied Region”  
End if

Pseudo code: Harris-SIFT Forgery detection System
4.3. Experimental Results and Discussion

The HARRIS-SIFT Forgery Detection System has been implemented using Matlab 7.6, in a computer of CPU 2.20 GHz with memory of 3 GB. The fast Harris detector along with SIFT descriptors are used to detect interest point and descriptors. The main task in any object recognition is matching the similarity between two feature points. For this KD-tree algorithm is used in the proposed system.

Since the image size is very important for any detection algorithms, six different images which are considered to be more challenging for copy-move forgery detection with different resolution and different size of copied area are used in our experiment. Three images are of high resolution of more than 2000 x 1600 pixels and three images are of low resolution. The copied region has basically the same appearance of the original one; therefore the keypoints extracted in the duplicated region will be similar to the original ones. Therefore, matching among the features can be adopted for the task of determining possible tampering.

Here, some experimental results on images where a copy-move attack has been performed. In this case the forged region is selected according to the specific goal to be achieved and, above all, paying attention to perfectly conceal a modification, where the alteration are not recognizable at least at the first glance and forensic tool could help to investigations. For instance the image Acropolis is forged with the right most statues which is marked with ellipse, the image Beachwood is forged with a green patch to conceal a building and the image. In the image Building two small statues on the left and right most broad pillars are copied and pasted in the second and third broad pillars. The crow and small green patch is copied and pasted in the image Cattle. Tree is modified with another tree and the neck marking in Giraffe is forged. The three tampered images Acropolis, Beachwood and Tree are affected with large area of forged region, while the images Building, Cattle (neck region) and Giraffe (neck region) is affected with small region of forged area. Further, in the image Building and Cattle more than one region is forged.
4.3.1. Threshold Settings for Forgeries Detection

In order to decide that two keypoints match (i.e. “are these two descriptors the same or not?”), simply evaluating the distance between two descriptors with respect to a global threshold does not perform well. This is due to the high-dimensionality of the feature space (128) in which some descriptors are much more discriminative than others. We can obtain a more effective measure by using the ratio between the distances of the closest neighbor to that of the second-closest one, and comparing it with a threshold.

For the Harris-SIFT Forgery detection System, the Harris threshold was taken as 300 for high resolution images (Acropolis, Beachwood and Building) and as 50 for low resolution images (Cattle, Tree and Giraffe). Interestingly, by decreasing the Harris threshold the number of keypoints increased, which resulted in more match points and subsequently detection time also increased. Since the resolutions of the images were different, the biggest challenge was to find the best threshold for matching in order to avoid false matching.

4.3.2. Test for Copy-Move Forgery Detection

a) Test on High Resolution Images

It is important to realize a fixed distance threshold cannot be used for all the images to evaluate the descriptors performance. Therefore, an in depth analysis was made to determine the best settings for the cut-off threshold $Dt$ (matching) for the high resolution images, which are shown in Table 4.1.

<table>
<thead>
<tr>
<th>Image</th>
<th>Threshold $Dt$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acropolis</td>
<td>0.15</td>
</tr>
<tr>
<td>Beachwood</td>
<td>0.06</td>
</tr>
<tr>
<td>Building</td>
<td>0.10</td>
</tr>
</tbody>
</table>
In Table 4.2 the number of keypoints extracted and the detection time (in seconds) are reported for three images with high resolutions for Harris threshold 300. It can be observed the Harris-SIFT Forgery System was able to detect large number of match points. For an example, for the image *Acropolis*, 1144 match points are obtained from 5339 key points extracted from the entire image. Similarly, for the image *Beachwood* the number of match points is 4132 from 18430 keypoints extracted. This is because large region in these two images are copied. On contrary for the image *Building*, the number of match points are less because the copied region is very small.

<table>
<thead>
<tr>
<th>Image</th>
<th>Harris Threshold($H_t$)</th>
<th>No. of Keypoints</th>
<th>Matches</th>
<th>Detection Time(Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acropolis</td>
<td>300</td>
<td>5339</td>
<td>1144</td>
<td>957.873</td>
</tr>
<tr>
<td>Beachwood</td>
<td>300</td>
<td>18430</td>
<td>4132</td>
<td>1293.143</td>
</tr>
<tr>
<td>Building</td>
<td>300</td>
<td>9494</td>
<td>103</td>
<td>2685.054</td>
</tr>
</tbody>
</table>

The keypoints extracted for each high resolution images *Acropolis*, *Beachwood* and *Building* and the results are shown in Figure 4.2, 4.3 and 4.4 respectively. The results indicate that the proposed method detects copy-move forgery efficiently.
Figure 4.2 shows the detected result for the tampered image *Acropolis*. The keypoints extracted are shown in the upper row and the forgery detection result is shown in the bottom row. A high number of matches are fundamental in order to identify the forged region. Although there are a number of sufficient match points for the image *Acropolis*, it can be noted that the entire patch of copied region (statue) was not identified (Figure 4.2 bottom row). However, compared to the FAST-SIFT Forgery System, the HARRIS-SIFT Forgery Detection System found more match points spread over the entire patch of the cloned area.
The Figure 4.3 shows the detected result for the tampered image *Acropolis*. The keypoints extracted are shown in the upper row and the forgery detection result is shown in the bottom row. It can also be noted the high number of keypoints are extracted from the image *Beachwood* for the given Harris Threshold $H_t = 300$. The system reliable detected the forged region with sufficient number of match points. It is interesting to note the entire patch of the cloned region was found by the system. From the result it is clear the Harris-SIFT Forgery detection System was far better than the FAST-SIFT Forgery System.
The Figure 4.4 shows the detected result for the tampered image *Building* using HARRIS-SIFT. The keypoints extracted are shown in the upper row and the forgery detection result is shown in the bottom row. It can be noted that the keypoints are extracted from most parts of the image. It could be observed a very small region is forged in the image *Building*. The lion from the left most pillar from the image is copied and pasted in the second pillar and similarly the lion from the right most pillar is copied and pasted in the third pillar from left. The System detected the entire region of the forged area much faster compared to the pervious method FAST-SIFT Forgery system.
Figure 4.5 shows the comparison between the numbers of keypoints extracted and keypoints matched for high resolution images. High numbers of keypoints were extracted from the image Beachwood.

![Figure 4.5: The numbers of keypoints extracted for high resolution images and the number of keypoints matched](image)

Figure 4.5: The numbers of keypoints extracted for high resolution images and the number of keypoints matched

Figure 4.6 illustrates the time taken to detect the duplicated region for high resolution image. It is interesting to note that the maximum time is taken by the tampered image Building. Although for the image Beachwood maximum number of keypoints extracted, it takes lesser time to match compared to the tampered image Building.

![Figure 4.6: Time taken to detect the duplicated region for high resolution images](image)

Figure 4.6: Time taken to detect the duplicated region for high resolution images
It can be observed from the Figure 4.6, the time taken to detect the forgery in each image. Although, the region cloned in the image *Building* is very small, the time taken for matching was higher than the other two images *Acropolis* and *Beachwood*. This mainly because of the textured nature of the image, the keypoints are extracted all over the image.

**b) Test on Low Resolution Images**

To analyze the performance of the proposed technique, the experiment was repeated with low resolution images. To determine the best settings for the cut-off threshold $D_t$ (matching) for the low resolution images was obtained after repeated analysis. The Table 4.3 illustrates cut-off matching threshold for the low resolution images.

<table>
<thead>
<tr>
<th>Image</th>
<th>Threshold $T_h$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cattle</td>
<td>0.08</td>
</tr>
<tr>
<td>Tree</td>
<td>0.13</td>
</tr>
<tr>
<td>Giraffe</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Table 4.3. The optimum matching threshold for the low resolution images.
In Table 4.4 the number of keypoints extracted and the detection time (in seconds) are reported for three images with low resolutions. It is interesting situation concerns the individuation of forged region for the image named *Giraffe* and *Tree*, the method able to detect a sufficient number of matched keypoints. On the contrary, for the image named *Cattle*, where two regions are forged, the method was able to detect only one forged region for the given Harris threshold (300). This is basically due to lesser number of keypoints extracted. Therefore, the Harris threshold was reduced to 50, to have sufficient number of keypoints for the low resolution images; the system detected both the cloned region effectively.

<table>
<thead>
<tr>
<th>Image</th>
<th>Harris Threshold</th>
<th>No. of Keypoints</th>
<th>Matches</th>
<th>Detection Time (Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cattle</td>
<td>50</td>
<td>4482</td>
<td>53</td>
<td>645.063</td>
</tr>
<tr>
<td>Tree</td>
<td>300</td>
<td>2274</td>
<td>37</td>
<td>242.517</td>
</tr>
<tr>
<td>Giraffe</td>
<td>300</td>
<td>2193</td>
<td>27</td>
<td>182.707</td>
</tr>
</tbody>
</table>

The keypoints extracted for each low resolution images *Cattle*, *Tree* and *Giraffe* are shown in Figure 4.7, 4.8 and 4.9 respectively. The results indicate that the proposed method detects copy-move forgery efficiently.
Figure 4.7: Forgery detection for the tampered image *Cattle* using HARRIS-SIFT.

The Figure 4.7 shows the detected result for the tampered image *Cattle*. The keypoints extracted are shown in the upper row and the forgery detection result is shown in the bottom row. As stated in the previous section, the system was able to detect only one forged region (patch of grass) from the image *Cattle* when $H_t$ was 300. Therefore, the Harris Threshold was reduced to 50 to extract sufficient number of keypoints. The system detected both the forged region (crow and patch of grass) and was more effective compared to FAST-SIFT.
The Keypoints extracted from the images Tree are shown in the upper row and the forgery detection result is shown in the bottom row in the Figure 4.8. The system was able to detect most of the region (branch) of the tree. However like the FAST-SIFT detection method this system too failed to detect the lower region of the tree (trunk).
Figure 4.9: Forgery detection for the tampered image *Giraffe* using HARRIS-SIFT.

The Keypoints extracted from the images *Giraffe* are shown in the upper row and the forgery detection result is shown in the bottom row in the above Figure. The system was able to reliable detect the forged region (near the neck) much faster than the FAST-SIFT method.
Figure 4.10 shows the comparison between the numbers of keypoints extracted and keypoints matched for low resolution images. Since the $H_t$ was considered 50 for the image Cattle, the number of keypoints were extracted was large compared to other two images. Since the forged regions on the low resolution images are small, the number of match points obtained was also less.

![Bar graph showing keypoints extracted and matched for Cattle, Tree, and Giraffe images](image)

Figure 4.10: The number of keypoints extracted for low resolution images and the number of keypoints matched.

Figure 4.11 illustrates the time taken to detect the duplicated region for high resolution image. It is interesting to note that the maximum time is taken by the tampered image Cattle.

![Line graph showing detection time for Cattle, Tree, and Giraffe images](image)

Figure 4.11: The time taken to detect the duplicated region for low resolution images.
4.3.3. Test on Multiple Copied Region

The proposed method was also analyzed to determine the performance of tampered images which have multiple copies of the same region. To address this problem one image form high resolution image (Acropolis) and another from low resolution image (Tree) was considered.

The right most statues from the image Acropolis copied and pasted in several different positions over the original image to create tampered image with multiple forgery of same region. The Left most image has two forged region, the middle image has three forged region and the right most image has four forged region as shown in top row. The detected results are shown in the bottom row of the figure 4.12.

Figure 4.12: Examples of tampered images (Acropolis) with multiple cloning of same region are shown in the first row and the detection results are reported in second row.
Table 4.5 illustrates the number of keypoints extracted, the number of keypoints matched and the detection time for Acropolis after multiple forgeries. The number of keypoints extracted and the number of keypoints matched increased proportionally when there was more number of forged regions. Similarly, the time taken to match also was on rise when forged regions are added.

<table>
<thead>
<tr>
<th>No. of Forgery</th>
<th>No. of Keypoints</th>
<th>Matches</th>
<th>Detection Time(Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two</td>
<td>6034</td>
<td>1760</td>
<td>1109.366</td>
</tr>
<tr>
<td>Three</td>
<td>6687</td>
<td>2388</td>
<td>1102.753</td>
</tr>
<tr>
<td>Four</td>
<td>7378</td>
<td>3017</td>
<td>1382.903</td>
</tr>
</tbody>
</table>

To create multiple forgeries in the image Tree, the tree region was copied and pasted in different part of the original image. The Left most image has two forged region, the middle image has three forged region and the right most image has four forged region as shown in top row. The detected results are shown in the bottom row of the Figure 4.13.

Figure 4.13: Examples of tampered images (Tree) with multiple cloning are shown in the first row and the detection results are reported in second row.
It is interesting to note that the number of keypoints and the number of matched points proportionally increased for the image Acropolis, where the lighting of the image is same throughout the image. On contrary for Tree with multiple forgeries the numbers of matched points are not same for the entire forged region because illumination throughout the Tree image was not same.

<table>
<thead>
<tr>
<th>No of Forgery</th>
<th>No. of Keypoints</th>
<th>Matches</th>
<th>Detection Time(Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two</td>
<td>2788</td>
<td>39</td>
<td>228.790</td>
</tr>
<tr>
<td>Three</td>
<td>2800</td>
<td>80</td>
<td>232.205</td>
</tr>
<tr>
<td>Four</td>
<td>2803</td>
<td>98</td>
<td>225.851</td>
</tr>
</tbody>
</table>

4.3.4. Test on Transformation-Scaling

In this section, the performance of the proposed system is analyzed to test images which have undergone some geometric transformation. Scaling and rotation belongs to geometric transformation type, which first relocates the positions of the image points, and then requires some sort of intensity interpolation to compute the image intensities at the new image points. The right most statute from the image Acropolis was copied and was subjected to geometric transformation (scaling) to obtain tampered images with different attack. Table 4.7 summarize the different geometric transformations attack applied to the cloned part in the image Acropolis. For an example in the attack F, the x and y axes are scaled by 120%. The table also results the number of keypoints extracted and keypoints matched for each attack.
Table 4.7: Different combinations of geometric transformation (scaling) applied to Acropolis

<table>
<thead>
<tr>
<th>Attack</th>
<th>$s_x$</th>
<th>$s_y$</th>
<th>No. of keypoints points</th>
<th>Matched Points</th>
<th>Detection Time (Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1.0</td>
<td>1.1</td>
<td>5340</td>
<td>714</td>
<td>1001.084</td>
</tr>
<tr>
<td>b</td>
<td>1.1</td>
<td>1.0</td>
<td>5416</td>
<td>742</td>
<td>1335.734</td>
</tr>
<tr>
<td>c</td>
<td>1.1</td>
<td>1.1</td>
<td>5379</td>
<td>487</td>
<td>991.971</td>
</tr>
<tr>
<td>d</td>
<td>1.0</td>
<td>1.2</td>
<td>5300</td>
<td>271</td>
<td>1040.291</td>
</tr>
<tr>
<td>e</td>
<td>1.2</td>
<td>1.0</td>
<td>5399</td>
<td>229</td>
<td>1107.052</td>
</tr>
<tr>
<td>f</td>
<td>1.2</td>
<td>1.2</td>
<td>5296</td>
<td>91</td>
<td>1020.944</td>
</tr>
<tr>
<td>g</td>
<td>1.0</td>
<td>0.9</td>
<td>5416</td>
<td>620</td>
<td>899.059</td>
</tr>
<tr>
<td>h</td>
<td>0.9</td>
<td>1.0</td>
<td>5376</td>
<td>658</td>
<td>973.504</td>
</tr>
<tr>
<td>i</td>
<td>0.9</td>
<td>0.9</td>
<td>5404</td>
<td>391</td>
<td>955.999</td>
</tr>
<tr>
<td>j</td>
<td>1.0</td>
<td>0.8</td>
<td>5434</td>
<td>125</td>
<td>955.440</td>
</tr>
<tr>
<td>k</td>
<td>0.8</td>
<td>1.0</td>
<td>5346</td>
<td>77</td>
<td>1278.195</td>
</tr>
<tr>
<td>l</td>
<td>0.8</td>
<td>0.8</td>
<td>5380</td>
<td>15</td>
<td>1405.964</td>
</tr>
</tbody>
</table>

The geometric transformations (scaling) attack applied to the cloned part in the image *Acropolis* and the experimental result obtained are shown in the Figure 4.14.
Figure 4.14: Forgery detection for different scaling
The figure 4.15 illustrates the number of key points extracted and the number of match points obtained for the selected scaling. For an example attack $k$ the cloned region the scaling factor $S_x, S_y$ applied to the x-axis and y-axis by 80%. For attack $f$ the cloned region the scaling factor $S_x, S_y$ applied to the x-axis and y-axis by 120%.

![Bar chart showing keypoints extracted and matched for different scaling factors](image1.png)

**Figure 4.15:** The performance measure for selected scale change

It is evident from the result that the number of match points decreases when the scaling factor $S_x, S_y$ applied to the x-axis and y-axis is increased or decreased.

![Line graph showing keypoints matched for different scaling factors](image2.png)

**Figure 4.16:** Match points for selected Scale change
4.4. CONCLUSION

A novel methodology to support image forensic investigation based on Harris Interest Point and SIFT descriptors has been proposed. Given a suspected photo with high resolution and low resolution, the system can reliably detect if certain area has been duplicated. Furthermore, the methodology can effectively detect a tampered image which has undergone transformation such as scaling. However, the system is weak in detecting images which has undergone attacks such as rotation, rotation & scaling and Gaussian noise which are addressed in the next chapter.