COPY-MOVE FORGERY DETECTION IN DIGITAL IMAGES

Ali Retha Hasoon Khayeat

July 2017

A thesis submitted
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

Supervisors: Dr Xianfang Sun, Professor Paul L. Rosin

School of Computer Science and Informatics
Cardiff University, United Kingdom

DECLARATION

This work has not previously been accepted in substance for any degree and is not concurrently submitted in candidature for any degree.

Signed ………………………………………………………… (candidate)

Date…………………………………………………….

Statement 1

This thesis is being submitted in partial fulfilment of the requirements for the degree of PhD.

Signed ………………………………………………………… (candidate)

Date …………………………………………………….

Statement 2

This thesis is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by explicit reference.

Signed ………………………………………………………… (candidate)

Date …………………………………………………….

Statement 3

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisation.

Signed ………………………………………………………… (candidate)

Date …………………………………………………….

Abstract

The ready availability of image-editing software makes it important to ensure the authenticity of images. This thesis concerns the detection and localization of cloning, or Copy-Move Forgery (CMF), which is the most common type of image tampering, in which part(s) of the image are copied and pasted back somewhere else in the same image. Post-processing can be used to produce more realistic doctored images and thus can increase the difficulty of detecting forgery.

This thesis presents three novel methods for CMF detection, using feature extraction, surface fitting and segmentation. The Dense Scale Invariant Feature Transform (DSIFT) has been improved by using a different method to estimate the canonical orientation of each circular block. The Fitting Function Rotation Invariant Descriptor (FFRID) has been developed by using the least squares method to fit the parameters of a quadratic function on each block curvatures. In the segmentation approach, three different methods were tested: the SLIC superpixels, the Bag of Words Image and the Rolling Guidance filter with the multi-thresholding method. We also developed the Segment Gradient Orientation Histogram (SGOH) to describe the gradient of irregularly shaped blocks (segments).

The experimental results illustrate that our proposed algorithms can detect forgery in images containing copy-move objects with different types of transformation (translation, rotation, scaling, distortion and combined transformation). Moreover, the proposed methods are robust to post-processing (i.e. blurring, brightness change, colour reduction, JPEG compression, variations in contrast and added noise) and can detect multiple duplicated objects. In addition, we developed a new method to estimate the similarity threshold for each image by optimizing a cost function based probability distribution. This method can detect CMF better than using a fixed threshold for all the test images, because our proposed method reduces the false positive and the time required to estimate one threshold for different images in the dataset. Finally, we used the hysteresis to decrease the number of false matches and produce the best possible result.

Acknowledgements

I am grateful to Allah, the Prophet Muhammad and Ahl al-Bayt (the peace and blessings of Allah be upon them) for giving me the strength and ability to attain this goal.

My sincere appreciation and gratitude go to my supervisors Dr. Xianfang Sun and Professor Paul L. Rosin for their continuous valuable advice and guidance, and their constructive feedback on my thesis. Thank you for all the skills I learned from you and thank you for being great supervisors.

I dedicate this work to my wonderful family. My lovely wife, my beautiful daughter, my mother, my amazing brothers and my aunt. It is a blessing to have you in my life.

I would like also to thank all the members of the School of Computer Science and Informatics at Cardiff University for their kind assistance.

List of Acronyms

ANMS	Adaptive Non-Maximal Suppression
ANN	Approximate Nearest Neighbour
AWGN	Additive White Gaussian Noise
BoWImage	Bag of Words Image
BRIEF	Binary Robust Independent Elementary Features
CHT	Circular Harmonic Transforms
CMF	Copy-Move Forgery
CMFD	Copy-Move Forgery Detection
CRMF	Copy-Rotate-Move Forgery
DCT	Discrete Cosine Transform
DLT	Direct Linear Transformation
DSIFT	Dense Scale-Invariant Feature Transform
DWT	Discrete Wavelet Transform
FAST	Features from Accelerated Segment Test
FFRID	Fitting Function Rotation Invariant Descriptor
FLANN	Fast Library for Approximate Nearest Neighbours
HOG	Histogram of Oriented Gradients
HOGM	Histogram of Oriented Gabor Magnitude
HSV	Hue, Saturation, Value
IMD	Image Manipulation Dataset
JPEG	Joint Photographic Experts Group
LBP	Local Binary Pattern
LBPFV	LBP Feature Vector

LSH	Locality-Sensitive Hashing
MAD	Median Absolute Deviation
MIFT	Mirror Reflection Invariant Feature
MLBP	Multiresolution Local Binary Patterns
MSER	Maximally Stable Extremal Regions
ORB	Oriented FAST and Rotated BRIEF
PCA	Principal Component Analysis
PCET	Polar Complex Exponential Transform
PCT	Polar Cosine Transform
PFP	Pixel False Positives
PHT	Polar Harmonic Transform
PNG	Portable Network Graphics
PST	Polar Sine Transform
RANSAC	RANdom SAmple Consensus
RGB	Red, Green, Blue
RIDSIFT	Rotation Invariant DSIFT
SATS	Same Affine Transformation Selection
SGOH	Segment Gradient Orientation Histogram
SIFT	Scale-Invariant Feature Transform
SLIC	Simple Linear Iterative Clustering
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
TIFF	Tagged Image File Format
UCID	Uncompressed Colour Image Database

Colour key

In this thesis, the matched blocks in the primary detection image have been highlighted by red line. The matched blocks in RANSAC image have been highlighted by a yellow line. The final detected images are highlighted with different colours for visualisation purposes and to show the correctness of CMF detection, see the table below.

F-measure	**Colour**
True Positive (TP)	Green
True Negative (TN)	Black
False Positive (FP)	Blue
False Negative (FN)	Red

Table of Contents

Chapter 1
Introduction

1.1 Introduction

An image can more strongly influence viewers than millions of words; images are used as evidence in courts, scientific research, political campaigns and celebrity magazines. Images represent a more natural and efficient way to communicate with humans than text does. For example, there is no need to translate images from one language to another. The rapid availability, ease of use and wealth of inexpensive devices to capture, store and send images (mobile devices, digital cameras and scanners) have helped to spread them. Simultaneously, the wide availability of software packages to edit images makes it very simple even for novice users to modify the image or create a new one. This increases the possibility of counterfeiting and tampering of visual data, which is no longer restricted to experts. As a result, the confidence and integrity that images once had is eroded by the advancement of digital technology. For instance, 100% of images in fashion magazines are retouched [1]. The topic of this research is about detecting one type of image tampering, the copy-move forgery.

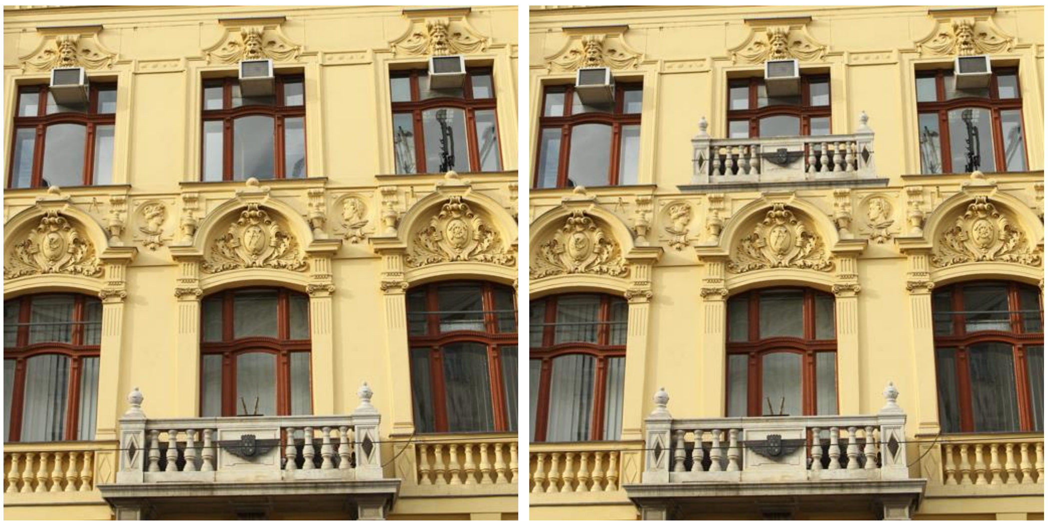

Figure 1-1: An example of copy-move forgery image; (left) the original image, (right) the tampered image [2].

1.2 Copy-Move Forgery

It is straightforward to create a copy-move forgery by cloning parts of the image to cover element(s) in the same image, but it is very hard to detect Copy-Move Forgery (CMF) by eyes when it is done by care, see Figure 1-1. The general CMF detection system consists of several main steps, see Figure 1-2. The first step is to pre-process the image, for example, by converting the RGB colour image to a greyscale image. The second step is to extract features from the image. There are two different methods of extracting them: dividing the image into blocks (densely); or detecting interest points in the image (sparsely). With the first method, the image can be divided into overlapping or non-overlapping blocks, which can be either square or circular in shape. The features are extracted from the blocks. In the second, the numbers and the locations of the interest points vary, depending on the method itself (e.g. Scale-Invariant Feature Transform (SIFT) [3], Speeded Up Robust Features (SURF) [4], etc.). The features are then extracted in the neighbourhood of the interest points. The third step is to find the matches (similarity) between the extracted features. Many methods can be used to locate these similarities. The most common method is either to sort the feature vectors lexicographically and compute the Euclidean distance between adjacent stored vectors [5]; or build a k-d tree containing all the feature vectors and find the 2nd Approximate Nearest Neighbour (2ANN) for each feature [6].

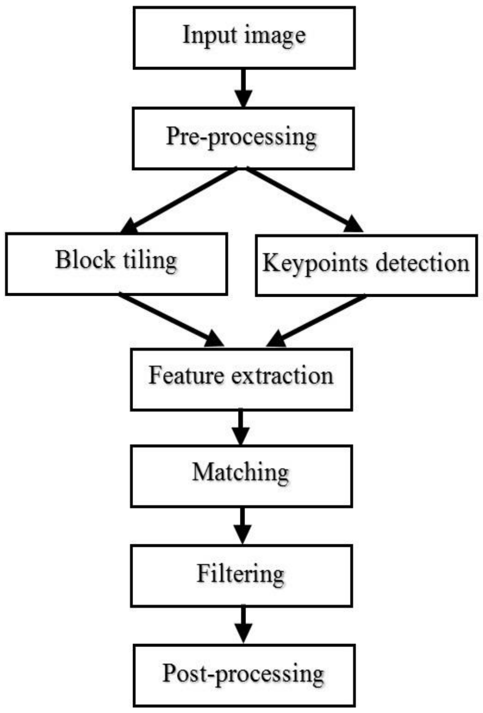

Figure 1-2: the general block diagram of the CMF detection system.

The pixel values/feature vector of the copied and rotated, scaled, or sheared part(s) are different from the original parts because of the interpolation, and these changes should be considered in the matching process. In the filtering, false matches should be removed to refine the primary result, followed by post-processing the result; for example, filling the holes in the large object and/or removing the small objects which are less than a threshold.

There is a big difference in the computational cost and amount of detected details between block-based methods and keypoint-based methods. Keypoint-based methods have the advantage of low computational complexity (they consume very little memory and are much faster than block-based methods). At the same time, keypoint-based methods cannot produce highly accurate results (because of detecting only parts of the Copy-Move objects or producing a false negative in flat regions).

1.3 Research Hypotheses

- Keypoint-based methods (e.g. SIFT, SURF) perform best in detecting image forgery with scaling and rotation. But to detect image forgery with Gaussian noise and JPEG compression, we have to use block-based methods such as DCT, DWT, and PCA. Therefore combining the elements of both methods into one system has a better chance of overcoming the problems of other suggested systems and being more accurate.
- There are major problems in the standard approaches to detect CMF (i.e. the block-based methods usually need a long time to extract from the image, while the keypoint-based methods can only detect part(s) of the duplicated objects). To overcome these problems, we propose to apply a segmentation-based approach. We hypothesise that over-segmentation methods (e.g. superpixel [7]) will be more appropriate than under-segmentation methods [8], because they allow a larger range of features (e.g. statistics) to be considered to describe each segment. The segmentation will divide the image much better than a block-based approach because it will exhibit boundary adherence, which can improve the accuracy of CMFD and reduce the required computation time.
- The greyscale image can be treated as a 2D surface, and it is possible to find clone regions by comparing the similarities over this surface. Consider that fitting surface method can be used to describe the properties of each block. Since the detection of Copy-Rotate-Move objects requires a rotation-invariant descriptor, we have to consider a rotation-invariant method in the description of each surface (e.g. by rotating the parameters of the fitted surface so as to be parallel to the principal axes).

1.4 Publications

1.4.1 Journal paper

A journal paper will re-submit soon to IEEE Transactions on Information Forensics and Security.

Khayeat, Ali Retha Hasoon, Xianfang Sun, and Paul L. Rosin. "Copy-Move Forgery Detection with Improved DSIFT Descriptor."

1.4.2 Conference Papers

- A. R. H. Khayeat, X. Sun, and P. L. Rosin, “Improved DSIFT Descriptor Based Copy-Rotate-Move Forgery Detection,” in *Image and Video Technology*, vol. 8334, no. November, F. Huang and A. Sugimoto, Eds. Berlin, Heidelberg: Springer International Publishing, 2016, pp. 642–655.

- A. R. H. Khayeat, P. L. Rosin, and X. Sun, “Copy-Move Forgery Detection Using the Segment Gradient Orientation Histogram,” vol. 10270, P. Sharma and F. M. Bianchi, Eds. Cham: Springer International Publishing, 2017, pp. 209–220.

1.5 Structure of the Thesis

The thesis is organized into six chapters describing the investigations of the research together with the main objectives of this doctoral study.

Chapter 1: a general introduction about the research subject, the problems and scope.

Chapter 2: a detailed description of the different types of doctored images and reviews in some depth the related methods and research.

Chapter 3: describes in detail our improved DSIFT descriptor and its use to detect CMF with different types of transformation and post-processing. It also presents the proposed method to estimate threshold automatically and our application of the hysteresis technique.

Chapter 4: describes in detail our novel method of using fitting surfaces of image blocks in CMFD with different types of transformation and post-processing.

Chapter 5: describes in detail three proposed methods (SLIC Superpixels, Bag of Word Image and Rolling Guidance Filter with Multi-thresholding) of detecting CMF using a segmentation approach. This chapter discusses the differences between our segmentation approach and those in other related studies.

Chapter 6: contains the conclusions, suggestions for future research and a comparison of our proposed methods.

Chapter 2
Review of the Literature in Image Forgery Detection

2.1 Introduction

Figure 2-1 is one of the earliest doctored images in the history, Abraham Lincoln's head was copied and pasted on to another politician's body (splicing). In Figure 2-2, Stalin's political enemy was removed from the image, which is another type of forgery (copy-move forgery). In the past, the task of doctoring images was extremely difficult and time-consuming as limited tools and devices were available at that time.

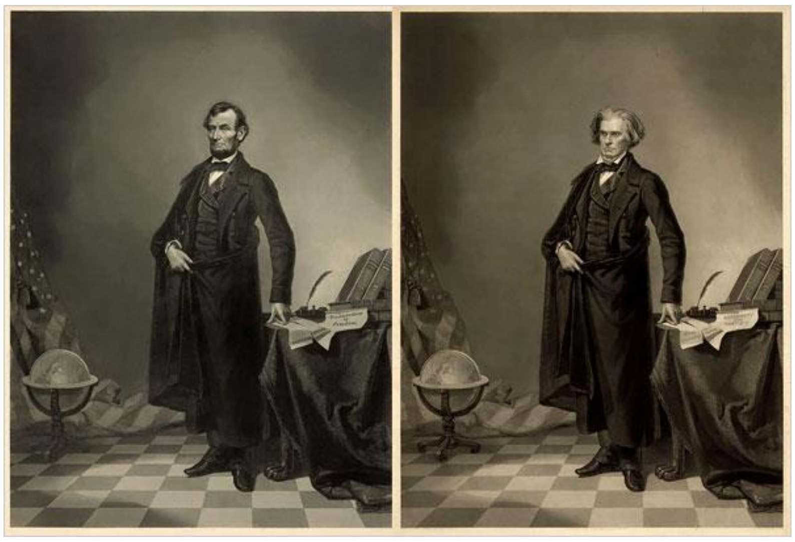

Figure 2-1: The portrait of Abraham Lincoln is composite of Lincoln's head and Southern politician John Calhoun's body [9].

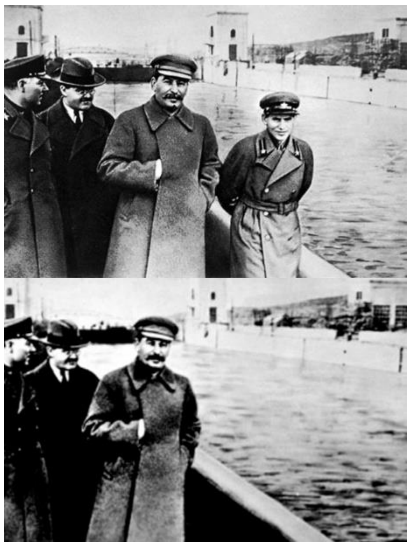

Figure 2-2: Stalin's political enemy was removed [10].

On the contrary, nowadays, little effort is needed to implement such task. Hence, the numerous affordable, easy-to-use and powerful digital image acquisition, processing, and editing devices and tools which are available to the end users. Therefore, both professional and amateurs can, easily and rapidly, alter images without leaving any visible trace. Moreover, the Internet makes the transmission of doctored images much easier and faster than before. The confidence in digital images have been lost, especially in sensitive data such as news items, medical records and evidence in court, etc. Particularly, the field of digital image forensics arose with the primary goal of developing efficient and reliable image forgery detection methods.

2.2 Digital Image Forensics

The digital image forensics, which is part of multimedia forensics, deals with:

- **Image source identification.**
- **Detection of computer generated images.**
- **Digital image forgery detection.**

The digital forgery detection methods can be classified into Active methods and Passive (blind) methods. Digital watermarking and digital signature are considered as active methods as they need to embed some information in the images before storing or transmitting. On the other hand, blind methods try to find whether the image is authentic or not without any previous embedded information [11].

2.2.1 Types of Digital Image Forgery

There are six main types of image forgery: Cloning, Splicing, Retouching, Morphing, Enhancing, and Computer Generating.

1. **Cloning:** Cloning or Copy-Move Forgery (CMF) is the most common type of image forgery, which is easy to implement and difficult to detect. In CMF, the forger copies part(s) of the image and pastes them back into the same image, see Figure 2-3. If CMF is done with care, its visual detection is difficult. Moreover, because the cloned regions can be in any location or can have any shape, searching all the possible image portions of different sizes and in different locations is computationally infeasible [11]. Many possible kinds of transformation (rotation, scaling, shearing and combining of several types) can be observed in forgery. For example in using rotation, the parts are Copied-Rotated-Moved (CRM) in the same image and a rotation-invariant feature must to be used to detect this type of forgery. Moreover, many possible post-processing manipulations can be suggested (e.g. adding noise, blurring, colour reduction, etc.) for making the doctored image look more realistic. Since the copied-pasted region is from the same image and the post-processing operation is done on the whole image, the characteristics of the copy-move region(s) (e.g. colour and noise) are compatible with that image. This type of forgery is harder to detect than other types, such as splicing and retouching. This is because the usual methods of detecting incompatibilities, using statistical measurements to compare different parts of the image, are useless for CMF detection [12].

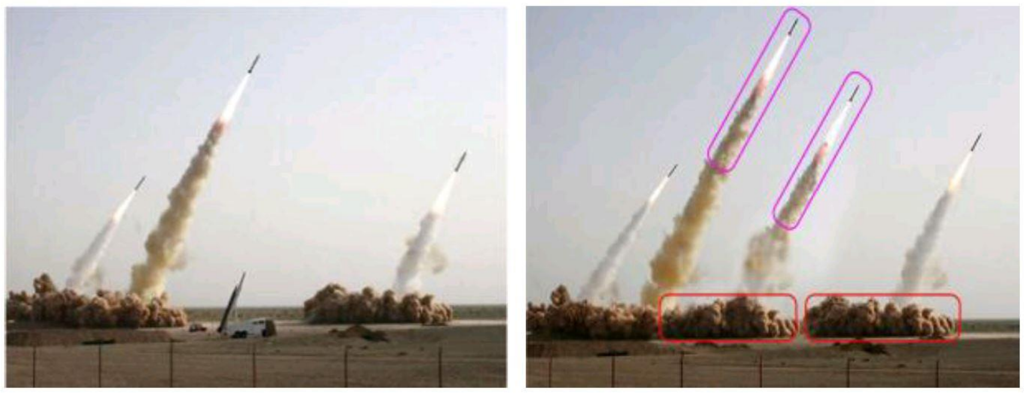

Figure 2-3: An example of copy-move forgery that appeared in the press in July, 2008.The original image (on the left) shows the original images and the tampered image (on the right) shows four Iranian missiles; two different sections (one encircled in red and the other in purple) replicate other image sections by applying a copy-move attack [12].

2. **Splicing:** Using a composite of two or more images to create a new one is a common type of photographic manipulation. When splicing is done carefully, the border between the spliced regions is sometimes visually imperceptible, see Figure 2-4. Such photomontages can be seen in several infamous news reports which include the use of faked images [11].

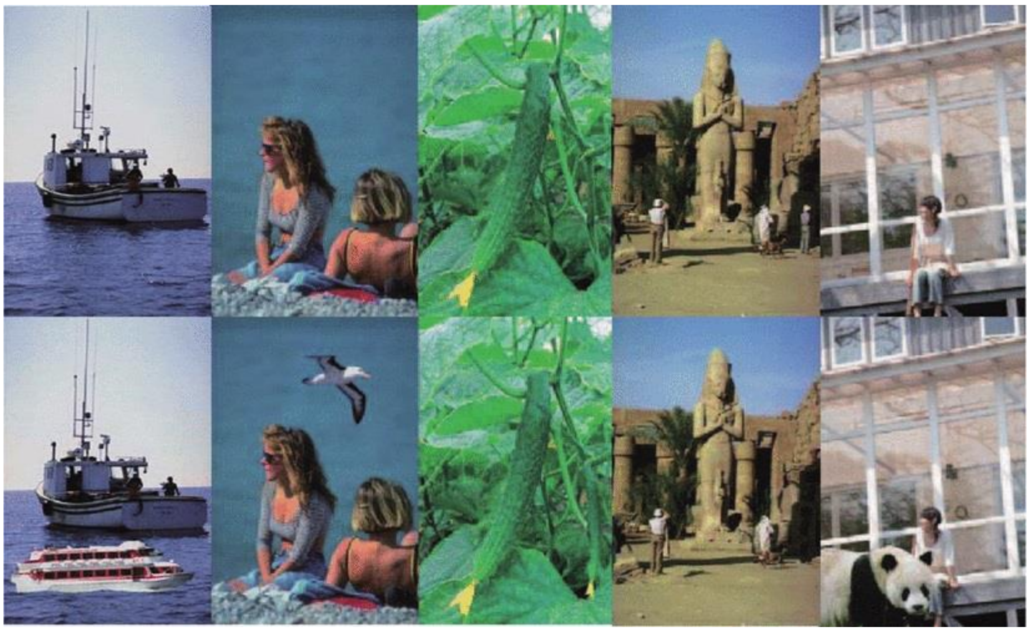

Figure 2-4 Examples of image splicing; all the images in the top row are authentic and those in the lower row are spliced [13].

3. **Retouching**: In the past, a film was retouched by painting over it with a very finely-pointed brush, using special dyes. Nowadays, digital retouching is much easier and quicker. As seen on Figure 2-5, an original image of an actor has been digitally retouched to make him look younger. This tampering involved copy-moving small patches to lower the hairline, remove wrinkles and remove the dark shadows under the eyes [14].

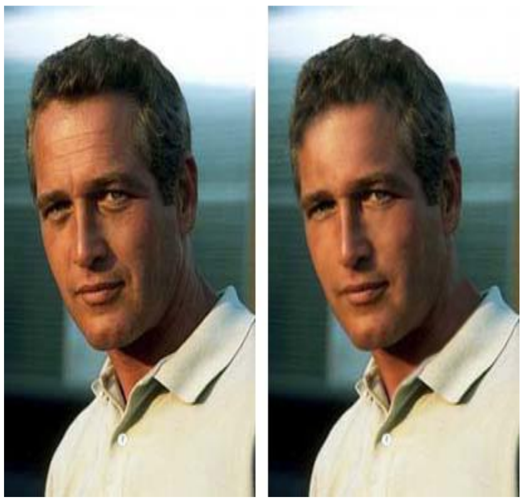

Figure 2-5: an original image of an actor (left), and a digitally re-touched image to make him seem younger (right) [14].

Retouching can also be used to repair damaged images, see Figure 2-6.

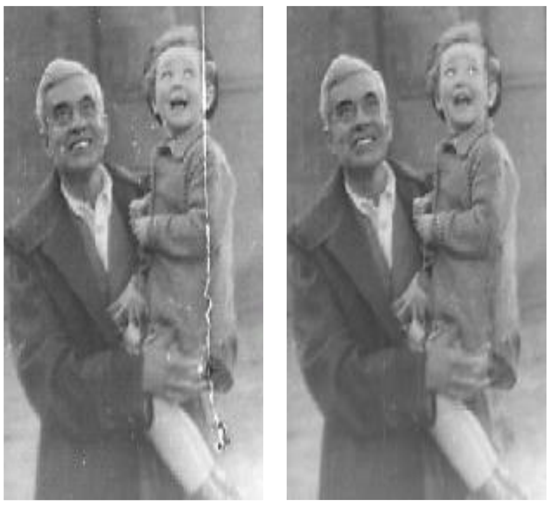

Figure 2-6: an example of image repairing [15].

4. **Morphing:** Image morphing is a digital technique that gradually converts one image into another. Shown in Figure 2-7, the image of a person (source image) is morphed into the image of an alien doll (target image). As shown, the shape and appearance of the source slowly change into the shape and appearance of the target. The intermediate images have features from both the source and target images, and have an aspect that is "part human, part alien" [14].

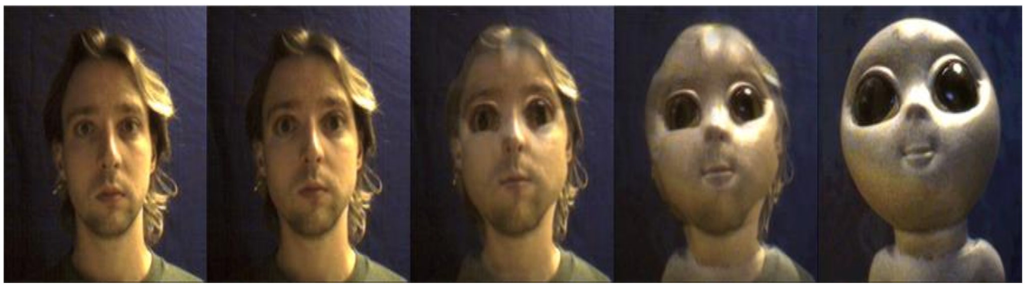

Figure 2-7: Example of a sequence of morphed images: a human face (the source) is slowly changed into that of an alien doll (the target) [14].

5. **Enhancing:** This type of tampering does not alter the content of the image but it includes contrast/colour adjustment, blurring and sharpening. Yet this type of tampering can still have an indirect effect on the interpretation of an image, such as altering the time of the day when the image appears to have been taken, see Figure 2-8 [14].

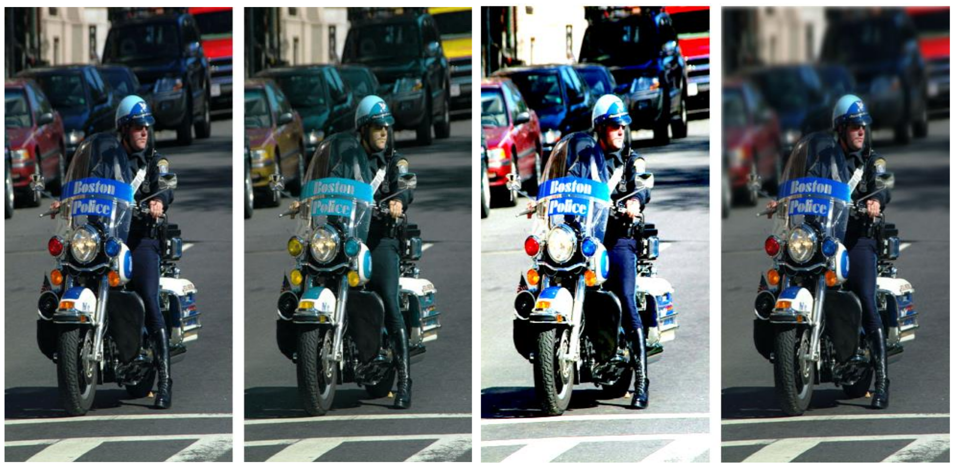

Figure 2-8: (left to right) an original image, the image enhanced to alter the colour, contrast, and blur of the background cars[14].

6. **Computer Generating:** a computer generated image can be defined as an image created by a skilled artist/programmer using a computer, whereas other types of image forgery (splicing, cloning, retouching, morphing, enhancing) alter the appearance of a photograph (either from a digital camera or a digitally scanned picture), see Figure 2-9 [14].

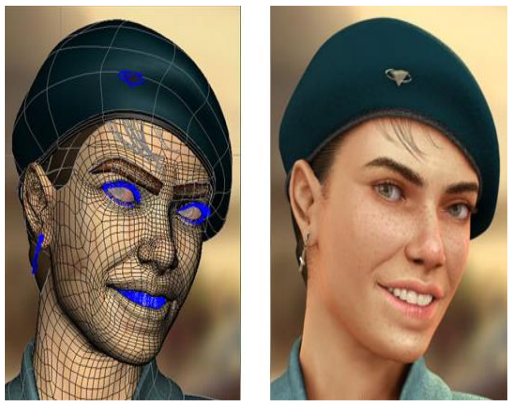

Figure 2-9: A computer generated model (left) and the resulting rendered image (right) [14].

2.2.2 Image Forensic Tools

According to [11], the image forensic tools are divided into five categories: Format-based techniques, Camera-based techniques, Physically-based techniques, Geometric-based techniques and Pixel-based techniques.

1. **Format-based techniques:** using statistical correlations in specific lossy compression algorithms to detect tampering in the images with JPEG format.
2. **Camera-based techniques:** using the camera lens, sensor or hardware (on-chip) postprocessing to detect tampering.
3. **Physically-based techniques:** these techniques use physical objects, the source(s) of light and the camera to create a 3-dimensional model for detecting anomalies [16].
4. **Geometric-based techniques:** measuring the objects in the world and their positions relative to the camera.
5. **Pixel-based techniques:** working at pixel level to detect statistical anomalies which relate to the scope of our work.

2.3 Image Forgery Datasets and Evaluation Methods

2.3.1 Image Forgery Datasets

1. **Columbia Image Splicing Detection Evaluation Dataset:** Ng and Chang [17] generated automatically spliced grayscale images dataset which contain 1845 image blocks. They copied part of the (128×128) image and pasted it randomly in a different image. Because they automatically created their images and did not consider post-processing, their images have sharp edges and are not semantically meaningful, see Figure 2-10.

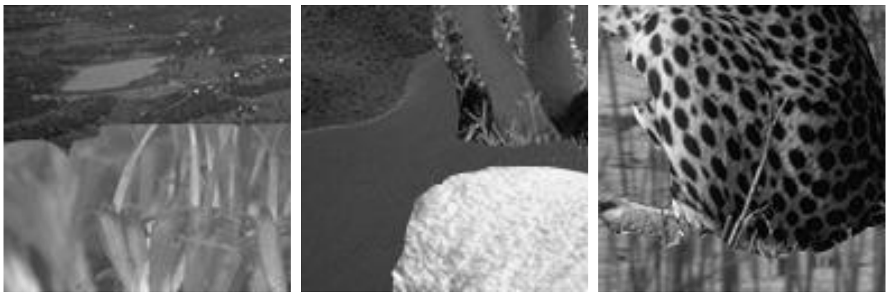

Figure 2-10: examples of the Columbia Image Splicing Dataset [17].

2. **The CASIA database:** There are two versions of the CASIA dataset [18]. CASIA v1.0 focuses on the detection of splicing; it consists of 800 authentic and 921 spliced colour images 384×256 pixels in size with a JPEG format. This dataset is divided into several categories (scene, animal, architecture, character, plant, article, nature and texture). CASIA V2.0 contains larger images than CASIA V1.0, and is more realistic because the tampered regions have been post-processed. It contains 7491 genuine images and 5123 tampered colour images. CASIA v2.0 has images of various sizes, from 240×160 to 900×600 pixels. It has uncompressed images and also JPEG images of different quality. In CASIA v2.0 the "indoor" category has been added to previous groups of images. The copied regions have been scaled and rotated before pasting. These images contain limited post-processing methods, only JPEG compression and blurring and most of the images are small (384×256).

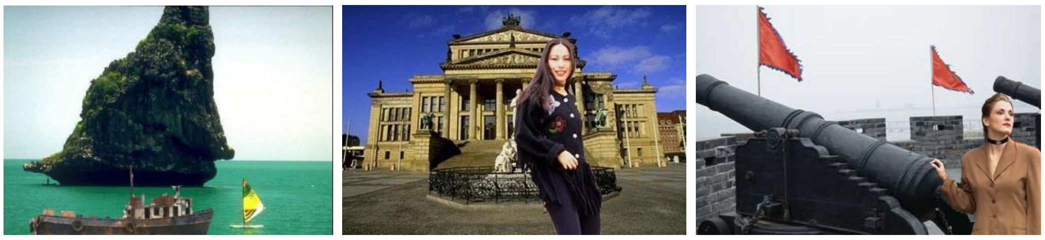

Figure 2-11: examples from CASIA images showing tampering very clearly [18].

3. **Databases MICC F220 and MICC F2000:** These datasets consist of 220 and 2000 images respectively. They build their forged images by randomly copying the location and the dimension of a square or rectangular area from the image and pasting the shape over the same image. Different types of transformation have been applied to the forged images, such as translation, rotation, scaling or combination [19].

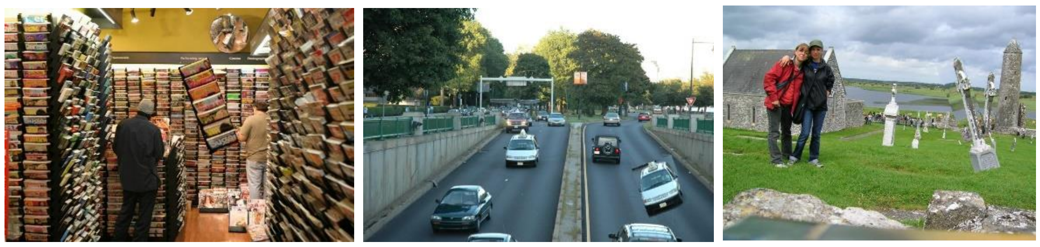

Figure 2-12: examples from MICC F220 images in which tampering is very clearly shown [19].

4. **CMFD benchmark Dataset:** Christlein et al.[20] created CMFD dataset in two steps:

   Step one: They selected 48 source images and manually prepared semantically meaningful regions from each image (snippets). They constructed 87 snippets with different content (e.g. smooth (sky), rough (rocks) or structured (man-made buildings)).

   Step two: They developed a software framework to produce copy-move images using these snippets. JPEG compression, noise adding, scaling or rotation are automatically included in the generated images.

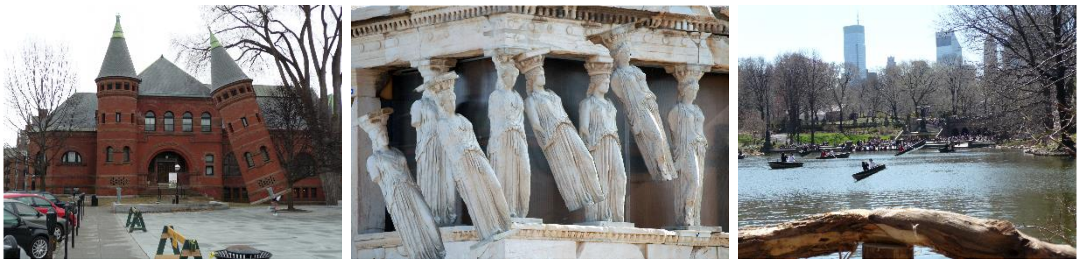

Figure 2-13: examples from the CMFD benchmark dataset with tampering shown very clearly [20].

Unfortunately, the tampering in some images of the CASIA, MICC and CMFD benchmarks is very obvious, making it needless to develop a system to detect it, see Figure 2-11, Figure 2-12 and Figure 2-13.

5. **CMFD database GRIP:** Cozzolino et al. [21] presented a CMFD database with 80 images, 768×1024 pixels in size. They built the copy-moved regions with arbitrary shapes, see Figure: 2-14. Moreover, noise adding, JPEG compression, scaling and rotation were considered in generating the forged images.

Figure: 2-14: examples from GRIP dataset [21].

6. **COpy-moVe forgery dAtabase with similar but Genuine objEcts(COVERAGE) :** Wen et al. [22] have recently published a new dataset with 100 original-forged image pairs. Each original image contains Similar-but-Genuine Objects (SGBs), and illumination changes have been made at region level. Most of the images have been taken from store shelves which have a limited

relationship to the CMF subject, see Figure 2-15. The two researchers stored their original and forged images in TIFF format, and did not consider JPEG compression or blurred images in their work.

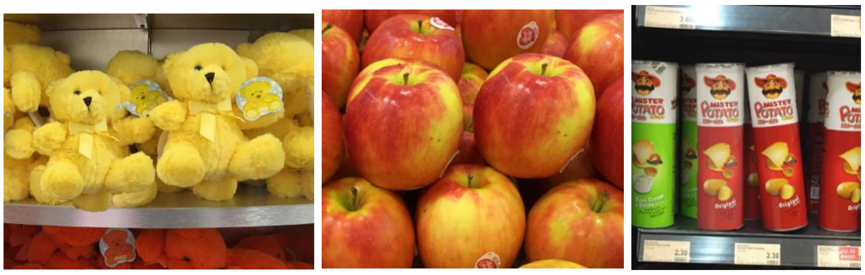

Figure 2-15: examples from the COVERAGE dataset.

7. **CoMoFoD - New Database for Copy-Move Forgery Detection:** CoMoFoD [23] consists of 260 forged images falling into two categories (small, 512 × 512, and large, 3000 × 2000). The small category consists of 200 original images demonstrating different types of forgery. In this category, images are divided into 5 different groups according to the manipulations applied, as follows: translation, rotation, scaling, distortion and a combination of all the previous manipulations. Moreover, various post-processing methods (e.g. blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise), are applied to all the forged and original images in each group. The total number of images in the small group is 10400, showing different types of manipulation. We consider this dataset (small images) in order to evaluate our algorithms because it allows more varied transformations and types of post-processing than other available datasets do.

2.3.2 Evaluation Methods

The F-measure is the most common method of evaluating the accuracy of the CMFD. There are two different types of F-measure: either at the image level or the pixel level. We used the F-measure [24] at the pixel level to evaluate the accuracy of our results.

$$\boldsymbol{Precision} = \frac{\boldsymbol{True\ Positive}}{\boldsymbol{True\ Positive} + \boldsymbol{False\ Positive}} \qquad \textbf{(2-1)}$$

$$\boldsymbol{Recall} = \frac{\boldsymbol{True\ Positive}}{\boldsymbol{True\ Positive} + \boldsymbol{False\ Negative}} \qquad \textbf{(2-2)}$$

$$\boldsymbol{F - Measure(F1)} = 2 \cdot \frac{(\boldsymbol{Precision} \cdot \boldsymbol{Recall})}{\boldsymbol{Precision} + \boldsymbol{Recall}} \qquad \textbf{(2-3)}$$

2.4 Copy-Move Forgery Detection Techniques Overview

A great many methods of CMF detection have been proposed, and they can be divided into three main types of techniques: block-based techniques [25] which performs very well but they need more time than the other methods. Keypoint-based methods [26] which have the advantage of low computational complexity, but they cannot produce highly accurate results. Segmentation-based methods [27] which cannot segment the identical objects consistently and depend on keypoint-based method to find the duplicated objects.

2.4.1 Block Based Techniques

To detect CMF in an image, an exhaustive search is the simplest approach. In this approach, the tested image and its circularly shifted version are overlaid to find the matched image parts, see Figure 2-16.

In this method, the equation (2-4) is used to examine the difference. Assume a grayscale image of size M×N and x_{ij} is the pixel value at the position i, j in that image.

$$|\, x_{ij} - x_{i+k\, mod(M)\ j+l\, mod(N)} | \qquad \textbf{(2-4)}$$

Where $k = 0,1, \ldots, M-1, l = 0,1, \ldots, N-1 \quad for\ all\ i\ and\ j.$

This approach is quite computationally expensive, taking $(MN)^2$ steps for an image of size M×N. Moreover, it cannot detect geometric transformed (e.g. rotated) copy-moved objects [25].

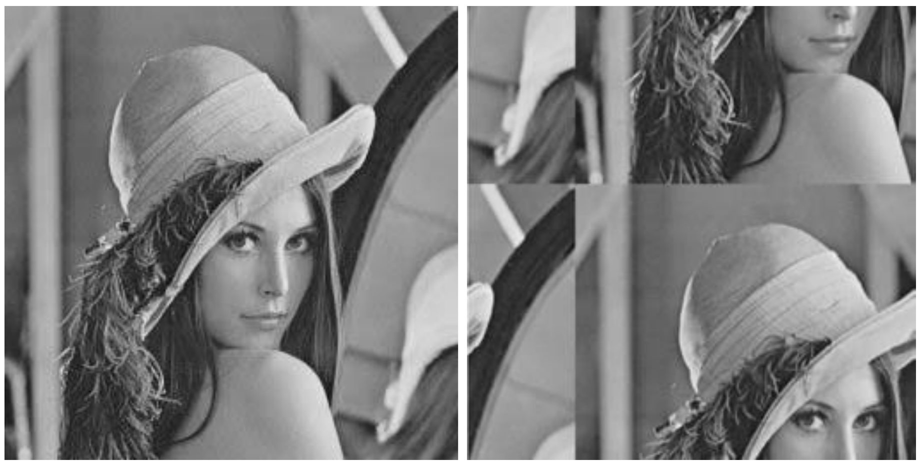

Figure 2-16: Test image "Lenna" and its circular shift[25].

a) DCT-based methods

The first method of detecting CMF was suggested by Fridrich et al. [25]. They divided the image into overlapping blocks and quantised the Discrete Cosine Transform (DCT) coefficients of each block; they then sorted them lexicographically and checked the similarity between adjacent blocks.

Hu et al. [70] divided the image into (8×8) overlapping blocks and computed the DCT coefficients from each block. In zigzag ordering, 8 coefficients are selected, according to frequency, from the array constituted by the quantized coefficients. The method is robust to blurring and noise contamination. However, in this work these researchers did not consider more complex transformation (e.g. rotation or scaling).

b) Statistical-based methods

Popescu and Farid [28] divided the colour image into overlapping (64×64) blocks and applied Principal Component Analysis (PCA) [29] on each channel. A matrix of quantized coefficients was constructed and sorted lexicographically. The offsets of the adjacent rows in the sorted matrix were computed. Then the coordinates of pairs with an offset frequency less than a certain threshold were removed. The coordinates of pairs with a distance of less than a threshold were also removed. The remaining pairs of blocks were used to build the duplication map. The authors used 100 colour images of size 512×512 pixels to test their algorithm. In each image, a random square region of 32× 32, 64× 64, 96× 96 or 128× 128 was duplicated. Each image was either JPEG compressed or corrupted with additive noise. They experimentally illustrated the robustness of their algorithm. However, they did not consider blurred images or more complex transformations (e.g. rotation, scaling) in their work.

Weiqi et al. [30] resized the images generated by Popescu and Farid [28] into 300×400 before using them. They converted the colour images into grayscale and used both representations in their work. They divided the colour/grayscale images into overlapping blocks 16×16 in size and extracted seven characteristics from each block. The first three characteristics are the average of the red, green and blue components of the block in the colour image. As shown in Figure 2-17, they divided each grayscale block into two equal parts (horizontally, vertically and diagonally) and computed the ratio between the sums of the two parts.

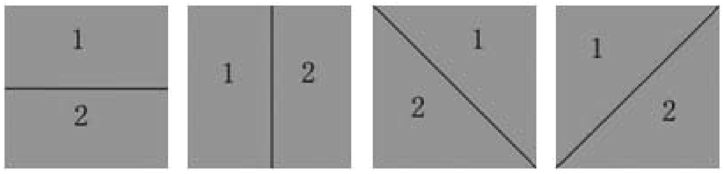

Figure 2-17: Division of the grayscale block [30]

The four generated values from each grayscale block are concatenated with the three features from the colour block. All feature vectors are sorted lexicographically and the similarity between block pairs is found using the difference between the characteristics of each block. They calculated the shift vector between each matched blocks. A histogram for the shift vectors was built to identify the highest frequency of occurrence and removed the other matched blocks. The indexes of the highest frequency are used to build a binary image which followed by applying morphological operations to remove the small areas and fill holes. They tested their algorithm with translation and different types of post-processing (e.g. JPEG compression, AWGN and Gaussian blurring). Their method is more robust against various post-postprocessing operations than Popescu and Farid [28]. However, they did not consider more complex transformations (e.g. rotation, scaling) in their work.

Lin et al.[31] used the green channel of the RGB image. They divided the image into 16×16 pixel overlapping blocks (B) and divided each block into four sub-blocks$(S_1, S_2\ , S_3\ , S_4)$. The average intensity for each block (f_1) and its sub-block (f_2, f_3, f_4, f_5) were computed. They then calculated the differences between the average of each sub-block (f_6, f_7, f_8, f_8) and the average of the main block(f_1) and normalized the nine numbers $(f_1, .., f_9)$ into integers between 0 and 255. The radix sort algorithm [32] was used to sort the list of the feature vectors. The shift vector, which is the difference between each adjacent feature vector in the sorted list and the cumulative number of shift vectors which are greater than a certain threshold were used to detect the duplicated regions, as they describe in their report. They then dealt with the CRMF with no interpolation, considering 90°, 180° and 270° in their work. The shift vectors could not detect forgery, see Figure 2-18. To overcome this problem, they combined three rotated versions of the tested image with the original one and tested the generated image.

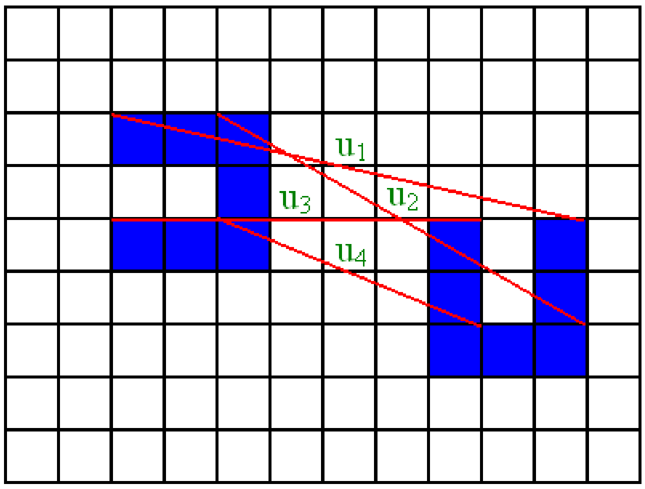

Figure 2-18: A region copied, rotated through 90° and pasted to another region [31]

They used their images to test their method. It should be noted that their method cannot detect forgery on rotated arbitrary angles, but this is a common problem in CMF.

Xu et al. [33] converted the colour image into grayscale and circularly shifted the image. For each circle shift, they divided the image horizontally, computed phase correlation for it and recorded any result that was higher than a threshold. Then, they divided the image vertically and repeated the same steps as before. Any sharp peak in the phase correlation that was higher than a particular threshold was used to detect forgery. They tested the performance of their method on some images from two data sets. Their method can detect forgery on images with translation and some postprocessing methods (e.g. blurring, JPEG compression and adding noise.) They did not consider multiple duplicated objects, rotation and the scaling in this work.

Lynch et al. [34] divided the image into overlapping 16×16 blocks. For each block, they computed the average grey value as the dominant feature. The blocks were sorted on the basis of the dominant feature into different groups. For each group, they constructed the connection matrix which denotes the blocks which are matched each other. Then, all the groups below a certain threshold in size were removed. The rows with zeros on the connection matrix were removed. Finally, they computed all the areas for the remaining connected blocks in each connection matrix and removed the small ones. According to their algorithm, the remaining blocks in the connection matrix represented the duplicated regions. They used 100 grayscale images 256×256 in size to test their algorithm. Their algorithm can detect forgery in images with translation, blurring and up to a 0.8 JPEG compression ratio. They computed their detection at an image level, and they did not consider rotation or the scaling in this work.

c) Transformation-based methods

Li et al.[35] converted the colour image into grayscale and filtered it using a Gaussian low pass filter. The filtered image was divided into overlapping circular blocks with a diameter equal to 16. The Polar Sine Transform (PST) [36], from Polar Harmonic Transform (PHT) [37], was employed to extract features from each block. The feature vectors were lexicographically sorted and each block feature compared with its adjacent 20 rows to find matches. Finally, morphological processing was employed to produce the final detection map. They tested the performance of their algorithm on images collected from the internet. Their method is robust to translation, rotation and some postprocessing methods (e.g. JPEG compression and adding noise). They did not consider scaling, blurring images, or the case of multiple copy-move forgeries in this work. Moreover, in their experiments they used images with a simple scene; the detection of the forgery in such images is much easier than in more complicated images, see Figure 2-19.

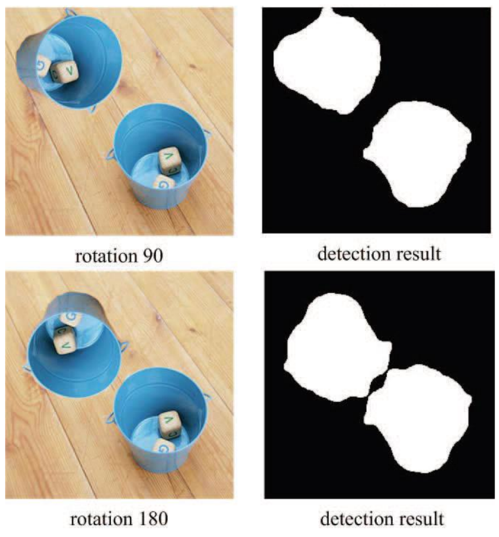

Figure 2-19: An example of forgery detection when the region is rotated by 90 and 180 degrees[35].

Shao et al. [38] divided the colour image into an overlapping circular block with a radius equal to 16. They used bi-linear interpolation to expand the circular block into a normalized rectangle block 16×128 in size. A Fourier transform was computed for each block and the researchers used an adaptive band limitation to obtain a correlation matrix with a suitable peak. They regarded the two circular blocks as being matched when the peak value calculated by band limitation phase correlation exceeded a specified threshold. A phase correlation method with no band limitation was implemented to estimate the rotation angle by the position of the peak in the correlation matrix. They then implanted a seed filling searching method to locate all the regions of forgery. They used 60 images from the image stitching database of the DVMM lab in Columbia University [17] and produced their forgery images using Photoshop. The method is robust to rotation, blurring, illumination changes, and JPEG compression. However, it is a time-consuming method owing to the point-by-point scanning in the circular block matching. The seed filling algorithm cannot detect some areas near a forgery boundary, and their proposed method is not robust against scaling.

Li [6] divided the image into overlapping circular blocks and extracted the Polar Cosine Transform (PCT) [37] from each block. Locality-Sensitive Hashing (LSH) [39] to find the approximate nearest neighbour patch was used to locate any matching between the feature vectors. A pair of patches is considered to be a possible forgery when its hash collides with high probability and the displacement vector between its coordinates is above a certain threshold. Finally, a duplication map is constructed to visualise the forgery detection result. The performance of their method was evaluated over 100 images with a squared duplicated area. Their method is robust to rotation, JPEG compression, blurred images and adding noise. It is considered that using forged images with a squared duplicated area makes forgery much easier to detect than irregular shapes do. These researchers did not consider scaling in their work.

Xiuli et al. [40] used a Multi-Level Dense Descriptor and Hierarchical Feature Matching to find the CMF. They converted the RGB colour into YCbCr colour space. Then, for each pixel, a colour texture feature and PCET moments [37] were computed. The PCET is one of the Polar Harmonic Transforms (PHTs) and is used in many applications, e.g. texture analysis and image registration. They grouped together all pixels with similar colour textures into specific neighbour pixel sets. They then used the PCET moments to match each pixel with its corresponding neighbouring pixel set. Finally, they set a threshold for the result and applied morphological operations to detect the forged regions. In their experiments, they scaled down by 50% the size of the tested images from two datasets. Consider that even after resizing the image, their proposed method takes about 37 mins to process one image. Moreover, the reason for converting the image from RGB space into YCbCr is not clear and nor what its effect on the CMFD is. Their method is robust to moderate rotation - rotating the duplicated object by 10° reduces the performance of their method by more than 25%, see Figure 10 in their paper. Finally, they did not consider blurred images in this work.

Wo et al. [41] extracted the Polar Complex Exponential Transform (PCET) [6] from the image. As the PCET is rotation invariant but not scale invariant, the authors extracted multi-radius PCET from each pixel, to generate multi-scale features. They used an improved lexicographical order matching method to find the copy-moved objects. They then filtered the mismatched points using SATS [42] and spatial information. They used the Image Manipulation Dataset (IMD) [12] to test their algorithm on images with plain CMF. They considered JPEG compression, blurring and adding noise in their experiments. Since the

IMD does not contain large-scale scaling and large-angle rotation forgeries, they used the Kodak lossless true colour suite (Kodak) [12] to duplicate a 100×100-sized patch in each image. They used multiples of 30° to rotate the square duplicated patch, and their method was dramatically affected by interpolation, see Figure: 2-20. Moreover, the performance of their method of detecting duplicated scaled objects depends on the consistency between the PCET radius in the multi-scale and the scaling factor.

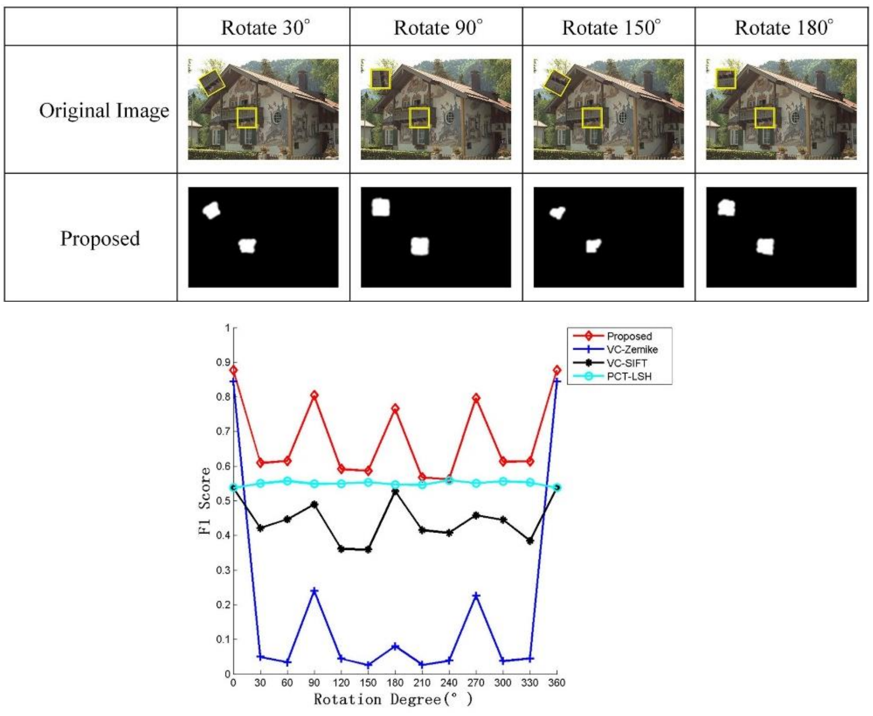

Figure: 2-20 Detection results against rotation operation and the F-measure [41]

Ketenci and Ulutas [43] smoothed the colour image by a mean filter and divided it into overlapping blocks. This feature extracted by applying the 2D-Fourier Transform on each block. The resulting feature vector was quantized and its dimensions reduced to 4 elements. The matrix of the features was lexicographically sorted. Each vector was compared, element by element, with its neighbours to find matching. This method is robust to translation, JPEG compression and image blurring, but it does not consider multiple duplicated objects or more complex transformation types (e.g. rotation or scaling).

d) Moments-based methods

Ryu et al. [44] divided the image into overlapping 24×24 blocks and calculated the Zernike moments for each block. They sorted the Zernike feature vectors lexicographically and computed the Euclidean distance between adjacent stored blocks. If the distance was below a certain threshold, they considered these blocks cloned. They conducted their experiments with 12 TIFF images from their personal collection and other papers. They considered copy-rotate move forgery with rotations in the range of 0°-90° in 10° steps. In their follow-up paper[45], they computed 5th order Zernike moments from overlapping blocks to generate their feature vectors. Locality sensitive hashing with Euclidean distances was used to find similar feature vectors. The authors applied RANSAC at the feature level to remove false matches. They then tested their work on their forged images, which were rotated between 0°and 90°, in 10° steps. They built their forged images by duplicating random square patches of different sizes on original images. This makes CMF/CRMF detection much easier but produces unrealistic forged images. Moreover, their method generates considerable Pixel False Positives (PFP).

Zhong and Xu [46] used Gaussian pyramid transform to extract the low frequency information from the image and divided it into overlapping blocks 8×8 in size. For each block, they used exponential-

Fourier moments and histogram invariant moments to build a 7-element feature vector. Dictionary sort was used to sort their features matrix. The duplicated regions were by thresholding the Euclidean distance between adjacent features. They did not consider postprocessing methods in their experiments (e.g. JPEG compression, blurred images or noisy images).

Cozzolino et al. [41] proposed a CMF detection algorithm based on a modified PatchMatch algorithm [42], for rotation-invariant and scale-invariant forgery detection. The Zernike moments and the RGB values extracted from 16×16 overlapping blocks were used as features. They evaluated the performance of their algorithm on 80 images from GRIP – the Image Processing Research Group [21]. Their method can detect forgery with rotation, scaling JPEG compression, and adding noise. They claimed that using Zernike moments increased the performance of detecting rotated duplicated objects by 30%. Unfortunately, they did not publish in their paper any images which show detected forgery. Moreover, they did not consider blurred images or the case of multiple copy-move forgeries in their work.

In their follow-up paper [21], focusing on Circular Harmonic Transforms (CHT) features[47] they replaced the use of pixel values in the modified Patch-Match algorithm [48] by scale and rotation features. They used Zernike Moments [49], Polar Cosine Transform (PCT) [37] and Fourier-Mellin Transform [50] instead of pixel values. They then applied a post-processing procedure to remove instances of false matching. They tested the performance of their method on two datasets [20] and a dataset which they had generated (GRIP) [21]. Although their method is robust to translation, rotation, moderate scaling and post processing methods (e.g. adding noise and JPEG compression), they did not consider blurring images in their work, which may reduce the efficiency of their algorithm.

e) Histogram-based methods

Lee [51] converted the colour image into grayscale. He divided the image into overlapped blocks (16×16) and applied Gabor filtering to each block. For each block, he computed a 12 bin Histogram of Oriented Gabor Magnitude (HOGM) descriptor. The feature vectors were lexicographically sorted. The matching blocks were found using the Euclidean distance between adjacent features. To reduce false matching, he removed the detected areas which were below a threshold size. He evaluated the performance of his method on two datasets, the CoMoFoD [2] and the Image Manipulation Dataset [20]. He considered rotation, scaling, JPEG compression, blurring, brightness adjustment and combination. His method can detect forgery with small angles of rotation and small scaling. Moreover, his work did not extend to noisy images.

Lee et al. [73] converted the colour image into grayscale and divided the image into overlapping blocks. They used blocks of three different sizes (16×16), (32×32) and (48×48), to discover the relationship between the block size and the forgery detection ratio. They then extracted the Histogram of Oriented Gradients (HOG) [12] from each block and sorted the generated feature vectors lexicographically. For each 5 neighbouring feature vectors, they computed the Euclidean distances to find the matched blocks. Finally, they removed the small detected areas which were less than 64 pixels in size. They used the CoMoFoD [12] to test their algorithm. Even though the HOG is not rotation invariant nor scaling, according to their paper they could detect forgery with very high correction results by translating small rotated objects and scaling. Moreover, they used lexicographical sorting which we found to produce errors in detecting CRMF because of the interpolation between the copy and rotated moved objects.

f) LBP and clustering methods

Davarzani et al. [74] created a database composed of 100 images from Columbia University dataset [65] and their personal collection. They resized the image into 260×260 pixels, converted the coloured image into grayscale and divided the image into overlapping blocks 10×10 in size. They filtered each block with Wiener filtering [13] to preserve the edges. They then extracted Multiresolution Local Binary Patterns (MLBP) [52] from each block. They used the k-d tree to find the 2ANN for each feature vector and refined their result using RANSAC. Their images were forged by copying a square region which makes the detection much easier than using irregular shapes and has a high detection ratio. The average run time of their method is greater than of any other block-based methods. Moreover, their method can detect forgery with rotation through limited angles especially when the angle is less than 20° or a multiple of 90°.

Hussain et al. [53] transformed the RGB image into YCbCr colour. They used the chrominance space to extract two local texture descriptors from the image: the multi-scale Weber's law descriptor (multi-WLD) [77] and the multi-scale Local Binary Pattern (LBP) [37]. They then used the Locally Learning Based (LLB) algorithm to remove the redundant features and reduce the dimension of the feature space. They used the Support Vector Machine (SVM) to find forgery at image level. Three data sets (CASIA v1.0, CASIA v2.0 [13], and Columbia colour [12]) were used to test the performance of their algorithm. The main drawback of their method, is that it cannot locate the forged objects on the image.

Akbarpour et al. [54] proposed a block matching method to avoid the feature matching of blocks belonging to the same area i.e. (sky, sea or grass). They focused on improving the time complexity of the block matching. They proposed a coarse-to fine block-matching model using a block clustering technique and local block matching. Their idea was to group similar blocks into a single cluster and then find matches by comparing the features of the blocks with each cluster, see Figure 2-21. They tested the performance of their method on MICC-F220 and other personal collected images. Their method can detect forgery better than lexicographical sorting on plain CMF but they did not consider rotation, scaling or postprocessing in this work. Moreover, they did not compare their method with the use of a k-d tree to find the nearest neighbour which, produces a better result with complex transformations than lexicographical sorting [55].

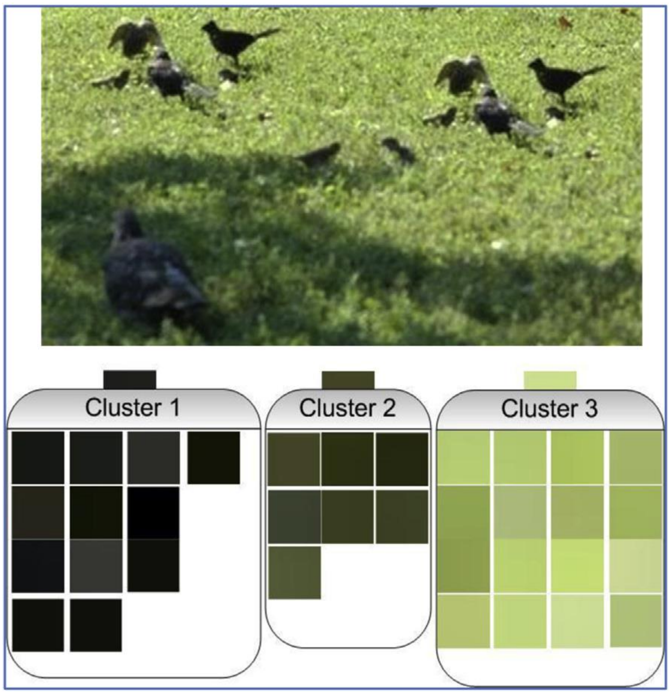

Figure 2-21: In the coarse-match step, all the image blocks are clustered on the basis of low accuracy features [54].

2.4.2 Keypoints Based Techniques

Huang et al. [26] extract SIFT keypoints from the image, and stored them in a k-d tree to ensure the efficient retrieval of the 2ANN. In their work, they used images from the internet. This method can partially detect CMF (one clone only), but they gave no consideration to post-processing methods, and the accuracy of their method is not reported.

Ardizzone et al. [56] extracted SIFT features, used a hierarchical tree to cluster keypoints and merged similar clusters. The SIFT feature descriptors of each cluster were used to match clusters. RANSAC was applied to remove the outliers and produce better matching results. They compared the content of the two matching clusters using the texture analysis and then thresholded their results. The algorithm gives acceptable results; the main weakness is that their method can detect copy-moved objects only partially. Moreover, they did not consider blurred images in this work and their method produced a false positive with translation, see Figure 2-22.

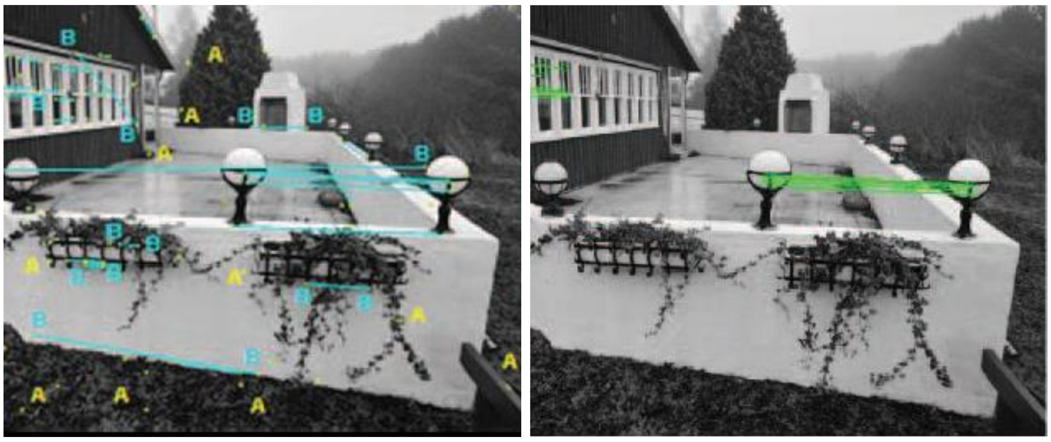

Figure 2-22: an example of using [56] to detect CMF, (left); an example of cluster matching (right) showing final result with false positive.

Pan and Lyu [57] detected sparse SIFT keypoints, and used the best-bin-first algorithm followed by RANSAC [58] to estimate possible geometric transformations. They built a correlation coefficient map between pixels in the same region and applied Gaussian filtering (7×7) to reduce noise, with a threshold imposed on the results. While using their own forged images, their method failed to detect forgery in some images with translation. Moreover, they falsely declared forgery in some original (untampered) images.

Amerini et al. [19] extracted SIFT features and used 2ANN to find multiple matches between feature vectors. They applied hierarchical clustering to their matched points and used RANSAC to estimate the geometric transform. The authors employed the Columbia photographic image repository [59] and personal collections of images. This method can partially detect multiple cloned regions, but it misses some objects and sometimes falsely detects forgery. Their algorithm cannot detect copied patches which have maximum uniform texture, such as the salient keypoints that are not covered by SIFT.

Shivakumar and Baboo [12] used a Harris corner detector to detect keypoints, which works faster than SIFT. They then used a SIFT descriptor to represent the extracted keypoints. The k-d tree algorithm was used to find matching and to detect duplicated regions. Their algorithm is weak when faced with Gaussian noise, scaling, and rotation.

Chen et al. [60] detected the Harris corner interest points [61] in the image. To detect enough feature points, this method has to increase the threshold which controls the number of Harris points for each single image. Then for each Harris point, a sector mask must be created with a radius of 20 pixels which is rotated by 10° every time. The mean and the standard deviation are computed for each sector and thus generated a 72-feature vector. Then the sector with the largest mean is taken as the direction mark of the circle block. The detected Harris points were matched on the basis of their feature vectors using the best-bin-first algorithm [62], with the distance between them necessarily greater than a certain threshold. Finally, all the matched points were displayed by circles of radius 20. To generate their forged images the authors used Adobe Photoshop with 24 uncompressed PNG images from the Kodak image database [63]. Squared regions of different sizes (60×60, 90×90 and 120×120) were copied and pasted on each image in their database. Their method can detect forgery on CMF and CRMF and is robust to JPEG compression and adding noise. However, for some images which contain flat regions, simply tuning for a global threshold will result in a many feature points leading to unnecessary computational cost, and unfortunately, the flat regions may still be not covered. Moreover, their method cannot detect scaled duplicated objects and can partially detect the copy-moved region, since the matched points are not clustered.

Insufficient SIFT keypoints, or even none, are found when the textures of some cloned regions are almost uniform (a flat region). To overcome this problem, Guo et al. [64] used the Adaptive Non-Maximal Suppression (ANMS) [65] to detect Harris interest points on the image. They then used an improved rotation DAISY [66] descriptor to represent each keypoint. To find the matching between two feature vectors, they used the ratio between the distance of the closest neighbour and the second-closest one and compared it with the threshold. To test their method, they randomly duplicated a square or rectangular area on 800 images from the Uncompressed Colour Image Database (UCID) [67]. Their use of a key-point based method made their method robust to rotation, scaling, JPEG compression and adding noise. Nonetheless, their method can partially detect the copy-moved region, because the matched points are not clustered. The use of ANMS to some extent solves the problem of detection keypoints in the flat regions. The ANMS was designed to obtain a fixed number of roughly uniform feature points according to the corner strength measure of each pixel location. Therefore, when an area is no more than roughly uniform (a flat region), the corner strength values are suppressed by the larger values around them, which produce a few or even no feature points. Moreover, these researchers did not consider blurred images in this work.

Jaberi et al. [68] used hysteresis thresholding to detect CMF. They determined the primary detection regions by finding the matches between sparsely extracted MIFT [69] features from the image. They then used RANSAC to remove outliers, and densely extracted MIFT features from each region. They used a fixed threshold (0.2) to find the primary matching between feature vectors. Next they tested the number of matches: if it was less than 10, they increased the threshold in steps of 0.05 until they had found 10 matches or reached a limit of 0.3. To perform primary detection, they chose the first threshold of hysteresis, depending on the number of matches. In addition, they centred windows (15×15) at keypoints and densely extracted the MIFT features from each window. They used the second threshold of hysteresis to test the similarity between corresponding feature vectors in each window. Their method could not detect forgery in flat regions because no interest point could be detected.

Amerini et al. [70] extracted SIFT features from the image and computed the Euclidean distance between each point. They used an adaptive version of the 2nd ANN to find the matching for each keypoint. The J-Linkage method [71] was used to merge the overlapping points with similar affine transformation into one cluster. They used normalised Direct Linear Transformation (DLT) [72] to estimate the affine transformation for each of the matched keypoints. The block-wise correlation was used to locate the duplicated regions, followed by applying morphological operations to fill holes in the results. They tested their work on three datasets (MICC-F2000, MICC-F600[19] and SATS-130[42]). Their method relies on SIFT which affects their detection on flat regions. Moreover, they did not consider any postprocessing methods in their work (JPEG compression, noisy images or blurred images).

Sliva et al. [73] converted the colour space of the tested image from RGB to HSV to decrease the generated false positives. The keypoints were detected on the V channel using SURF [4] and described with Haar Wavelet [74]. The Nearest Neighbour Distance Ratio was used to find matching points. They clustered the matched points according to their correspondence angles. A Gaussian pyramid decomposition was applied on the generated image to produce a partial detection map for each scale. Their method cannot detect the interest points on the flat regions and they did not consider postprocessing (e.g. blurring, adding noise, etc.) in their work.

Ardizzone et al. [75] used SIFT, SURF [4] and Harris [61] to find interest points on each image. A Delaunay triangulation [76] was built on the extracted points, which subdivided the image into triangles. For each triangle, each colour channel was quantized into 8 bins to make a 3D histogram. The most frequent values of the histogram were considered the dominant colours of the triangle. Starting from the maximum angles, these researchers computed the triangle areas and inner angles in a counter-clockwise direction. They segmented the image into triangles and described each triangle by its dominant colours, areas and ordered sequence of angles. The triangles were sorted according to colour and the primary matching triangles were found using the Sum of the Absolute Deviation between colour features. Then the centroids for matched tringles were computed and RANSAC was applied to remove outliners.

The authors used their own dataset for their major evaluation; their method can detect parts of copy-move objects as shown in Figure 2-23. However, they cannot detect forgery on homogenous areas and the performance of their method deteriorates on images with complex scenes.

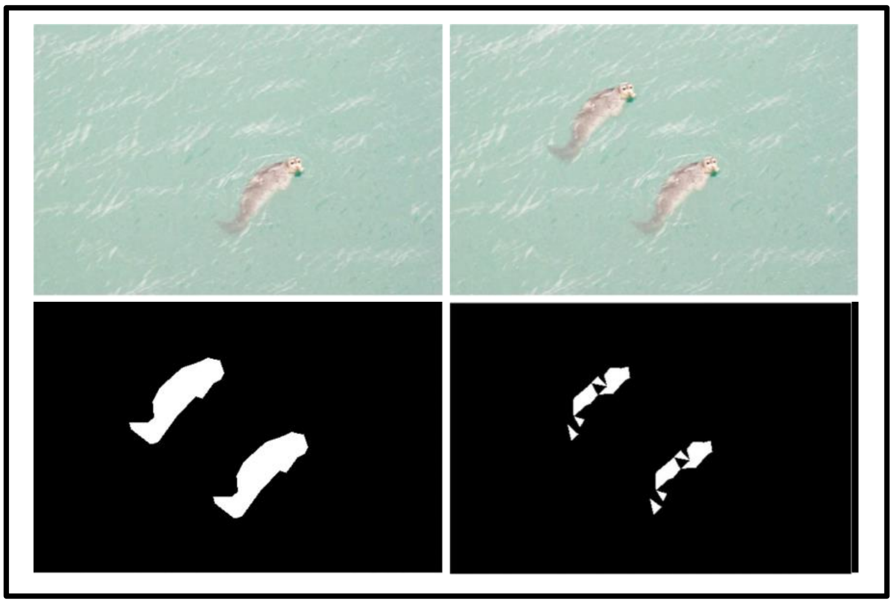

Figure 2-23: (top to button, left to right) Original image, Forged Image, Ground truth, Detected copy-moved areas [75].

Xhu et al. [77] established a Gaussian scale space [3] and extracted the oriented FAST keypoints [78] and the ORB [79] feature from each scale. The Hamming distance was used to find the matching between the descriptors, before RANSAC was applied to remove the false matches. These researchers evaluated their method on images from Columbia University natural images collection [59] and from the internet. They did not cluster their detected objects and therefore their method can detect only parts of copy-moved objects.

Many methods of detecting CMF have been suggested. Christlein et al. [80] tested the 15 most prominent feature sets by creating a real-world copy-move dataset and a software framework for systematic image manipulation. They analysed the performance of the detection on a per-pixel basis and a per-image basis. In their experiments, SIFT and SURF keypoint-based features work very well, and so do block-based DCT, DWT, kernel PCA, PCA and Zernike moments.

There is a great difference in computational cost and the number of detected details between block-based methods and keypoint-based methods. Keypoint-based methods have the advantage of low computational complexity (consuming very little amount of memory and much faster than block-based methods). Such methods, however, cannot produce highly accurate results (detecting only parts of copy-move objects or producing false negatives in flat regions).

2.4.3 Segmentation Based Techniques

The authors of [81] tested four different image segmentation methods and used superpixels Simple Linear Iterative Clustering (SLIC) algorithm [7], to over-segment the images. They then extracted the SIFT features from each segment, built a k-d tree for them and used the KNN to find the matching between patches. They computed the number of matched feature points and determined the suspicious pairs of patches which had many similar keypoints, applying RANSAC to estimate the transformation matrix between each pair of patches. They then used the dense SIFT descriptor to represent each pixel in all the matched patches and found the matching between the descriptors. Next they applied RANSAC to remove the outliers. Depending on their approach, they found that the method of segmentation does not greatly influence copy-move forgery detection. Their approach is similar to that taken for locating the matching between different images using SIFT features, considering that each segment as a different image. They used the segmentation method to divide the original image into various patches (small images). Their approach depends on extracting SIFT features at the level of segment instead of the image as a whole. Their method generated a high level of false positives with plain copy-move forgery.

Li et al. [27] also used the SLIC method to segment the images. They used different sizes of segmentation depending on the image content itself. They began by setting a large size for superpixels, with smooth images, and a small initial size with detailed images. Discrete Wavelet Transformation (DWT) was used to analyse the frequency distribution of the image. They found with this approach that the image is smooth when low-frequency is detected in most of the frequency energy of the host image; otherwise, the image is detailed when the low-frequency energy accounts for a minority of the frequency energy of the host image. They then extracted the SIFT features from each segment and computed the Euclidean distance between features. The number of matched points were calculated and generated the correlation coefficient map to find the matched patches. They used the SLIC to segment each matched patch to a smaller size and measure the local color feature for each sub-patch. They merged the neighbouring sub-regions (patches) when the color features were similar and applied morphological close operation to generate the regions where forgery could be detected.

Bhanu and Kumar [82] used SLIC to segment the image into more than 100 patches, and extracted the SURF [4] from each patch. They then compared the feature vectors of each patch with the rest using Approximate Nearest Neighbours and thresholded their result. They used RANSAC to estimate the transformation matrix for the copy-moved patches and removed the outliers. They then used Adobe Photoshop and some images from a benchmark dataset with the MICC-F600 dataset to create their forged images. It was found that the benchmark dataset and the MICC-F600 dataset have their forged images and there was no need to used Adobe Photoshop to create others. In spite of the SURF is rotation and scaling invariant, their method could not detect rotated copy-move objects or scaled ones. They did not explain the JPEG compression ratio or specify the amount of noise that had been added to their tested images. Moreover, they did not consider blurred images in their work.

Sekhar and Shaji [83] presented a study taking a segmentation based approach and rotation invariant DAISY descriptors to detect copy-move forgery. In their study, they used three existing methods; they followed the same approach as [27] to segment the image. They proposed to use Adaptive Non-Maximal Suppression (ANMS) feature detection and DAISY descriptors [64] instead of SIFT. Finally, they found the matching between features using the g2NN approach [19]. Since they did not implement their proposed method, there is no evidence whether or not their method would work.

2.4.4 Open research issues

As far as we know, none of the previous work has considered possible distortion (vertical elongation, horizontal elongation, vertical skew and horizontal skew) or post-processing (Colour reduction, Contrast adjustments) in image manipulation. This highlighted research ideas that are developed in the following chapters which considered CMF detection in images with distortion and postprocessing.

Chapter 3
Improved Dense Scale-Invariant Feature Transform for Copy-Move Forgery Detection

3.1 Introduction

As noted above, many methods of detecting CMF have been suggested. Christlein et al. [80] tested the 15 most prominent feature sets by creating a real-world copy-move dataset and a software framework for systematic image manipulation. They analysed the performance of the detection on a per-pixel basis and a per-image basis. In their experiments, SIFT and SURF keypoint-based features worked very well in detection CMF, and so did block-based DCT, DWT, kernel PCA, PCA and Zernike moments.

According to Christlein et al. [80], the Zernike moments achieved the most precise detection results (state of the art). Therefore, we compared our improved DSIFT with Zernike moments to see which produced better results.

One of our contributions is to have combined ideas derived from the keypoint and block-based methods. We chose the Scale Invariant Feature Transform (SIFT) method and applied it densely to make block-based matching possible. SIFT is the most widely used descriptor; it is distinctive and relatively fast. However, in some cases, the high dimensionality of the descriptor is considered a drawback in the matching step [26]. The other main contributions are improved the DSIFT and developed the automatic similarity thresholding.

3.2 Related Techniques

The techniques that were used in the proposed algorithm are as follows:

3.2.1 Scale Invariant Features Transform (SIFT)

In 2004 David Lowe presented a new algorithm, Scale Invariant Feature Transform (SIFT) [3], to extract keypoints and compute its descriptors. Four main steps are involved in the SIFT algorithm:

I. Scale-Space Extrema Detection:

SIFT uses Difference of Gaussians as a scale-space filter to make the SIFT scale invariant. The Difference of Gaussian, $D(x, y, \sigma)$, is found as the difference of Gaussian blurring of an image with two different standard deviations; let them be σ and kσ. This method was applied for different octaves of the image in the Gaussian Pyramid.

Once the DoG is computed, images are searched over scale and space to find local extrema. Each pixel in an image is compared with its 8 neighbours as well as 9 pixels in the previous scale and 9 pixels in the next scales. If it is a local extrema, it is a potential keypoint. This basically means that the keypoint is best represented in that scale.

II. Keypoint Localization

Once the locations of potential keypoints are found, they must be refined to produce more accurate results. SIFT uses a Taylor series expansion of scale space to produce more accurate locations of extrema. If the intensity at this extrema is below a specific threshold value, it is rejected.

The edges also need to be removed because the DoG has higher responses on edges. For this, a 2×2 Hessian matrix is used to compute the principal curvature at the location and scale of the keypoint. If the DoG is below the ratio of the largest to the smallest eigenvalue, from the 2×2 Hessian matrix at the location and scale of the keypoint, the keypoint is rejected. This step eliminates any low-contrast keypoints and edge keypoints, leaving only strong interest points [3].

III. Orientation Assignment

The orientation is assigned to each keypoint to achieve invariance to image rotation [3], this approach will explain in details in Section 3.3.1.

IV. Keypoint Descriptor

In this step, the keypoint descriptor is created. A 16×16 neighbourhood around the keypoint is taken. Each block is divided into 16 sub-blocks of 4x4 size. For each sub-block, an 8-bin orientation histogram is created thus generating a 128-element feature vector for each block [3].

Finally, the vector is normalized to set the maximum value of its elements to 255 and quantized to an 8-bit integer [84]. The SIFT descriptor is invariant to translations, rotations and scaling transformations in the image domain. Moreover the SIFT descriptor is robust to moderate perspective transformations and illumination variations. This descriptor has been used in many applications, e.g. in image matching and object recognition in real-world conditions.

3.2.2 Dense Invariant Features Transform (DSIFT)

Bosch et al. [85] and Dalal [86] suggest computing the SIFT descriptor on dense grids (Dense SIFT). The DSIFT has been shown to produce better performance than SIFT for tasks such as object categorization and texture classification. A basic explanation for this is that a larger set of local image descriptors computed over a dense grid usually provides more information than the corresponding descriptors evaluated over a much sparser set of image points [85].

The available implementation of DSIFT (e.g. VLFeat Dense SIFT) is not rotation or scale invariant; it is equivalent to applying SIFT on a dense gird of locations at a fixed scale and orientation.

3.2.3 Zernike moments

The regular moments can be defined as the projection of the $f(x, y)$ function onto the monomial $x^p y^q$. Consider that the basis set $x^p y^q$ is not orthogonal. Therefore, the recovery of an image from these moments is difficult and computationally expensive. Furthermore, the m_{pq} holds a certain amount of redundant information.

Zernike moments can be used as rotation invariant features, since rotating the image does not change the magnitude of its Zernike moments. In addition, it is easy to reconstruct an image from Zernike moments because the orthogonality property allows the individual contribution of each order moment (information content) to be separated. These individual information items of content can be used to reconstruct the image.

Consider that Zernike moments are rotation invariant only; to make them scale and translation invariant, the image must be first normalized using regular moments.

In image processing, Zernike moments carry out a mapping function of an image onto a set of complex Zernike polynomials [87]. Since these Zernike polynomials are orthogonal to each other, Zernike moments can characterize the features of an image with no redundancy or overlap of information between the moments [88][49]. With these characteristics, Zernike moments have been used as feature sets in applications such as content-based image retrieval [89], pattern recognition [88], and other image analysis systems [90][91].

As noted above (Chapter 2 several papers have used Zernike moments in CMF detection [44] [45][21]) .

3.3 Improved Dense Invariant Features Transform (DSIFT)

3.3.1 Steps to Improve DSIFT

On the basis of local image properties, SIFT assigns a dominant orientation for each keypoint. When building the descriptor, each patch is rotated according to this orientation so that the subsequent descriptor is robust to rotation [3]. CRMF detection requires a rotation-invariant descriptor; thus, we improved the DSIFT descriptor to make it rotation invariant. We improved two aspects of the DSIFT descriptor: first, we used a different method to compute the dominant orientation, and, second, we used circular blocks instead of square ones.

Dominant Orientation

SIFT uses the following approach to detect the dominant orientation for each patch. For each keypoint, compute the gradient orientations in its 16×16 neighbourhood. An orientation histogram containing 36 bins covering 360° is built. Each value added to the histogram is weighted by its gradient magnitude. Peaks in the orientation histogram represent the dominant directions of the keypoint. The highest peak in the histogram and any other local peak within 80% of the highest peak are used to represent the dominant orientation for the keypoint. Thus with multiple peaks of similar magnitude, multiple keypoints with different directions are created at the same location.

As Lowe explains [3], only about 15% of the keypoints are assigned multiple orientations, and this step significantly increases the stability of matching. Finally, to produce a more accurate result the peak in the orientation histogram is interpolated with the closest bins.

The standard setting of SIFT assigns multiple orientations with multiple peaks of similar magnitude, which causes problems at the CMF stage. It is possible to choose one orientation for each patch, but this can significantly reduce the stability of matching. Obviously, there is a trade-off between robustness and the multiple orientations approach.

We used two methods to detect the dominant orientation.

1. Detect the canonical orientation using the central moments

We used the second order and third order central moments to detect the canonical orientation. This method is more accurate and faster than the SIFT's method for detecting the dominant orientation. The second order central moment (moment of inertia) can be used to detect the principle axes of the patch, the region around the keypoint. The angle of the principle axis of the least inertia is used to describe the object orientation. This angle has a 180° ambiguity; the third central moment (projection skewness) is used to resolve this ambiguity. The rotation of an object by 180° changes the sign of the projection's skewness on either axis. In other words, the sign of $\mu_{30}$was used to differentiate between the possible orientations [92]. This method works very well and is much faster than the SIFT method, but it still takes a long time and many computations to find the canonical orientation for all the patches in the image.

The formulae to find the angle of the principle axis of least inertia is

$$\theta = \frac{1}{2}\arctan\left(\frac{2\,\mu_{11}}{\mu_{20}-\mu_{02}}\right) \qquad \textbf{(3-1)}$$

Every patch is rotated by the estimated angle θ and 180° is added to θ when μ_{30} is less than zero.

$$\text{Where}\quad \mu_{30} = m_{30} - 3\bar{x}m_{20} + 2\bar{x}^2 m_{10} \qquad \textbf{(3-2)}$$

2. Detect the canonical orientation using intensity centroid measure

Instead of the third order central moments, standard moments (intensity centroid measure) [93], were used to find the canonical orientation for each patch in the image.

The moment can be defined as follows:

$$\mu_{pq} = \sum_{x,y} x^p \, y^q \; I(x,y) \qquad \textbf{(3-3)}$$

The centroid can be determined as:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right) \qquad \textbf{(3-4)}$$

Consider that O is the origin and the canonical orientation is the angle of the vector $\overrightarrow{oc}$.

$$\theta = arctan\left(\frac{\mu_{01}}{\mu_{10}}\right) \qquad \textbf{(3-5)}$$

This method (intensity centroid measure) [93] is much faster than the third order central moment and gives the similar results. It reduces the computing time needed to find the canonical orientation and gives an accurate estimation of the orientation for all patches in the image. The proposed method has no quantization error or problem of multiple orientations. Moreover, Xu et al. [94] have found that centroid based orientation can estimate orientation more accurately and with better noise resistance ability than gradient based orientation.

Circular Blocks

SIFT considers a square region around the keypoint, which increases the border effects on this region. In spite of using a circular shaped Gaussian weighting in the standard SIFT can reduce the edge effects of square blocks, but it cannot totally eliminate the edge effects as circular blocks do. Our experimental results showed that the performance in forgery detection is sensitive to the edge effects. The simple explanation is that the multiple orientations approach makes the detection of CRM forgery more difficult and increases the number of false matches. This is simply because the bigger the interpolation change, the bigger the threshold value needed to distinguish similarities between feature vectors, and the higher the false matching ratio.

Instead of square area, we considered a circular area to reduce the border effects. Each block within a radius of 7.5 was divided into 4×4 sub-regions. A comparison between circular and square neighbourhoods is described in Section 3.3.3.

3.3.2 High-level description of the proposed algorithm to compute improved DSIFT

The steps in building our improved DSIFT descriptor are summarized as follows:

1. Transform a colour image into greyscale it is a colour image.
2. For each pixel in the image, consider its 16×16 neighbourhood.
3. Mask each neighbourhood to use only the central disk with a radius equal to 7.5.
4. Use the method based on the intensity centroid measure to find the canonical orientation for each circular patch.
5. Rotate each circular patch according to its canonical orientation.
6. Compute the gradient magnitude and orientation for each circular patch.
7. Use the Gaussian function to weight the gradient magnitude.
8. For each 4×4 sub-region in the circular patch, build an 8-bin gradient magnitude of orientation histogram.

9. Accumulate each bin according to its gradient magnitude of orientation.
10. Concatenate the 16 histograms to build a 128-element feature vector.
11. Normalize the feature vector between 0 and 1.

3.3.3 An Experiment to Test the Rotation Invariance of the improved DSIFT:

In Section3.3.1, we described how the level of rotational invariance of the DSIFT descriptor was improved. We also conducted an experiment to test our descriptor. For 40 different forgery images, we randomly selected 100 blocks from each image and computed our improved DSIFT descriptors for these blocks. Next, we randomly rotated these blocks, considering all the possible rotation angles (0°–360°), and computed our improved DSIFT descriptors for these rotated blocks. Then we computed the Euclidean distance between the descriptors of the original and the rotated blocks. The average Euclidean distance between 4000 pairs of improved DSIFT descriptors, built from 4000 different blocks before and after rotation, was 0.0487, see Figure 3-1.

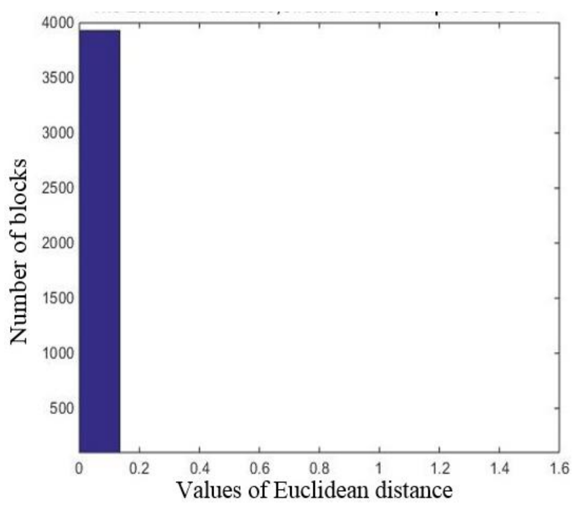

Figure 3-1: Histogram of the Euclidean distance between 4000 improved DSIFT descriptors.

The standard DSIFT is not rotation or scale invariant, see Section 3.2.2. We built a rotation invariant DSIFT using the standard SIFT method to estimate the dominant orientation and we called it Rotation Invariant DSIFT (RIDSIFT). To compare the robustness of our improved DSIFT rotation with the RIDSIFT, we repeated the previous experiment using the RIDSIFT. The average Euclidean distance between 4000 pairs of descriptors (square block), built from 4000 different blocks before and after rotation, was 0.8787, see Figure 3-2.

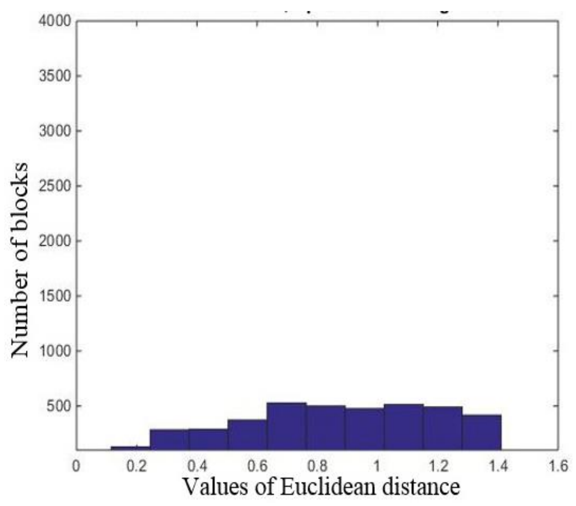

Figure 3-2: Histogram of the Euclidean distance between 4000 RIDSIFT descriptors with square blocks.

We then repeated the same experiment using the RIDSIFT with circular blocks instead of square ones; this time the value of the average Euclidean distance was 0.3396, see Figure 3-3. It was obvious that our version of DSIFT (improved DSIFT) was more robust to rotation than RIDSIFT.

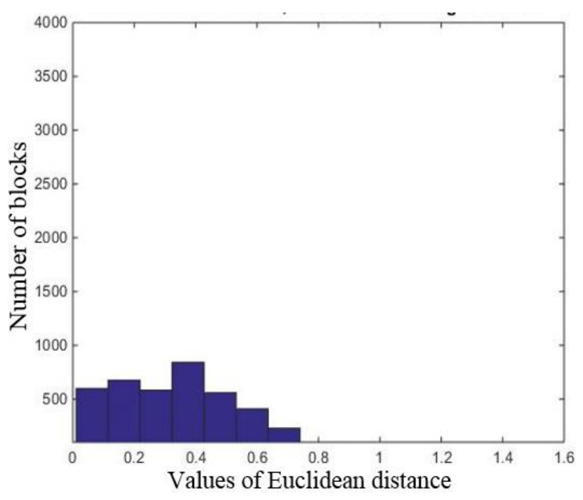

Figure 3-3: Histogram of the Euclidean distance between 4000 RIDSIFT descriptors with circular blocks.

3.4 Automatic Similarity Thresholding

The similarity threshold between feature vectors is one of the most important parameters in detecting CMF. This threshold depends on the image itself and is different from one image to another. The image characteristics (e.g. the texture, the colour distribution, and the edges) influence the similarity threshold.

The most common scenario in selecting the similarity threshold is as follows: Set a primary threshold to test all images in the training dataset, and then change (decrease, increase) this threshold, depending on the initial result of using it. Choose the single threshold which produces the best results, e.g. the highest F-measure, for all the images in the dataset. The major drawbacks of this approach are that it is time-consuming, and that it gives one fixed threshold for many different images. Obviously, various datasets need different thresholds and therefore it is necessary to repeat the optimization scenario. This substantially increases the evaluation time.

It is useful to have a new fast method of estimating the similarity threshold for each image separately.

Here, we propose to find the threshold by optimizing a cost function based on two probability distributions: one corresponds to the correct matching of a patch with its rotated and scaled counterpart, and the other is related to the false matching of different patches.

Suppose R(t) and V(t) are respectively the correct and false matching distributions related to threshold t. Then the cost function is defined as the estimated probability H(t) of correct classification in forgery detection for a given threshold t,

$$H(t) = \int_0^t R(t)dt + \int_t^\infty V(t)dt \qquad \textbf{(3-6)}$$

The optimal threshold T corresponds to the maximum value of H(t), i.e.,

$$T = \underset{t}{argmax}\ H(t) \qquad \textbf{(3-7)}$$

Figure 3-4 shows an example of the R(t) and V(t) distributions, and the corresponding optimal threshold.

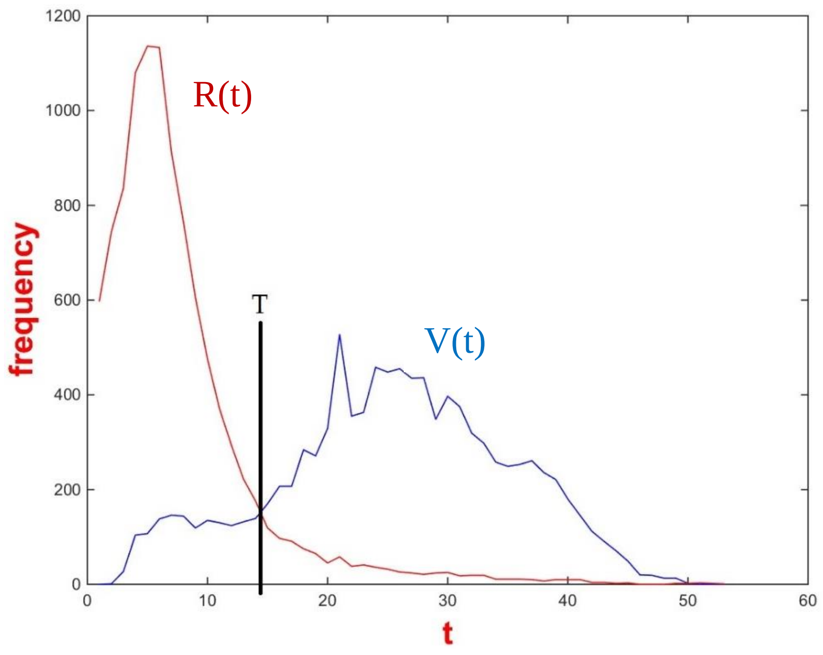

Figure 3-4: an example of plotting histograms R and V.

In principle, R(t) and V(t) should be continuous functions. However, we can get only a few discrete points to represent the distributions, so the optimal threshold T will be computed on the basis of the discrete numbers.

The first distribution, R(t), represents the distribution of the errors caused by interpolation during rotation and scaling. To estimate this distribution, we conducted the following experiment. Consider that the image size is (M×N); the block size is (B×B) and K = ((M-B)×(N-B) /10). For each image in the dataset, we randomly selected K patches, and computed the improved DSIFT descriptor for each patch. We randomly rotated each patch by a value in the range (0°-360°) and computed the improved DSIFT descriptor for the rotated patch.

Then we randomly scaled each patch by factor (0.5-1.5) and computed the improved DSIFT descriptor for the scaled patch. We computed the Euclidean distance between the feature vector of the original patch and the feature vector of the rotated/scaled patch and saved it in list {R}.

The second distribution V(t) is the distribution of mismatching blocks, and to estimate this distribution we randomly selected K pairs of patches from the image. We computed the improved DSIFT descriptor for each patch, and then computed the Euclidean distance between the two feature vectors and saved it in list {V}.

The two lists of distances {R} and {V} were binned to build histograms R and V, respectively. The optimal threshold was then simply computed on the basis of the histograms. That is, H(t) was computed as the sum of the cumulative histograms of R(t) from 0 to t and of V(t) from t to the maximum of t.

To determine the number of bins in the histograms, we use Scott's rule [95]. We wanted the two histograms to have the same bin width. The bin width was calculated by means of the following formula: $h=3.49\sigma/\sqrt[3]{n}$, where h is the bin width, n is the number of elements in each list (here, equalling K), and σ is the average value of the standard deviations of lists {R} and {V}.

The proposed method is fast. It required only ~9 seconds to find the optimal threshold for an image of size 512×512.

3.5 The proposed algorithm for detecting CMF using improved DSIFT

Bosch et al. [85] illustrated the DSIFT can perform better than SIFT in texture classfication. Using dense SIFT, instead of standard SIFT, increases the running time but provides robust features which are systematically distributed over the whole image. For example, computing DSIFT for a 512×512 image with a block size of 16×16 generates 247009 feature vectors. Computing sparse SIFT for the same image size typically generates about 750 to 1350 keypoints/feature vectors.

3.5.1 High-level description of the proposed algorithm to detect CMF using Improved DSIFT

The algorithm for copy-move forgery detection is as follows:

We converted the colour image to grayscale. Flat regions increase false matches. Such flat regions occur where the pixel intensity values are similar to each other and change smoothly over comparatively large regions (e.g. sky, sea, etc.). The similarity between pixel intensity values in a large region produces a large number of similar feature vectors, which are considered to be copy-move regions in the matching step. We used the Median Absolute Deviation (MAD) to reduce the effect of flat regions. If a block's MAD value was larger than the threshold, we built the improved DSIFT descriptor to the tested block; otherwise, we rejected it. This threshold had been optimized experimentally on the tested dataset. The proposed method reduced the number of false matches in the flat region(s) and cut the run time significantly.

We built the k-d tree for all feature vectors and used the Fast Library for Approximate Nearest Neighbours (FLANN) [96] to find the 2nd Approximate Nearest Neighbour for each feature vector.

The Approximate Nearest Neighbour is used since the interpolation and postprocessing operations add some changes to the copy-moved blocks/feature vectors. Consider the 1st Approximate Nearest Neighbour refers to the feature vector itself, the 2nd Approximate Nearest Neighbour is the coped-moved one within a specific threshold.

We computed the Euclidean distance between each vector and its 2nd Nearest Neighbour and saved feature coordinates when their distance was less than the weak threshold. We applied Neighbourhood Clustering to reduce the false matches and called the RANSAC to remove the outliers. We employed hysteresis to grow the primary detection and recoloured the matched blocks.

The proposed algorithm to detect CMF using improved DSIFT is described in full detail as follows:

1) Convert the image to grayscale if it is a colour image.
2) Use the proposed method in section 3.4 to compute the OptimumThreshold.
3) Divide the image into overlapping blocks (16×16)

   If the Median Absolute Deviation (MAD) of the tested block is greater than $\text{Threshold}_0 = 0.09$.

   i. Compute the feature vector of the improved DSIFT descriptor of the tested block.

ii. Save the feature vector and the coordinates of the tested block.

End

4) Build a k-d tree for the feature vectors and find the 2nd Approximate Nearest Neighbours for each element in the tree.
5) Find the matched feature vectors (blocks) which satisfy:
   a. The Euclidean distance between feature vectors is less than $\text{Threshold}_1 = (\text{OptimumThreshold}/2)$.
   b. The distance between the centres of the two matched blocks is less than $\text{Threshold}_2 = 16\sqrt{2}$.
6) Save the coordinators of the two matched blocks in a List.
7) Use the neighbourhood clustering method, proposed in Section 3.7, to remove the false matching from the List.
8) Call RANSAC to estimate the transformation of the List.
9) Use the hysteresis technique, proposed in Section 3.8, to grow regions of the List.
10) Remove small areas which less than (Threshold_3=320 pixels) in size.
11) Mark the blocks as matching.
12) Morphologically close the image.

3.6 False Matching Removal

We tested the following three methods to remove potential false matches:

1) **Counting shift vectors:** This method involved creating a list of coordinates for each potential cloned patch and sorting it. Then we computed the shift vector (spatial distance) between each related point. If the number of each of the shift vectors was greater than

the threshold, the patches were considered to be a forgery. This method is appropriate in the case of translation but not that of CRMF detection.

2) **Neighbourhood Clustering:** The copied and pasted blocks each had to comprise at least three neighbouring blocks within a radius. This method produced a very good result; details of the neighbourhood clustering are given in Section 3.7.

3) **RANSAC:** This is an iterative method of estimating the parameters of a mathematical model from a set of observed data which contains outliers. In the initial stage. RANSAC uses a dataset which is as small as possible; it consistently enlarges this dataset whenever possible. RANSAC can be used to robustly estimate the geometric transformation between matched points and remove outlier blocks [58]. It can cope with more than 50% of the outliers, making it more robust than many other parameter estimation technique (such as the least median of squares [97]). Figure 3-5 shows an example of using RANSAC for CMF detection to remove false matching.

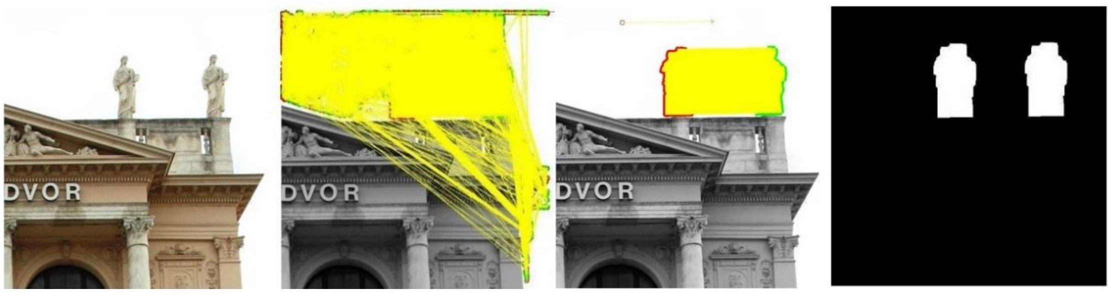

Figure 3-5: from the left: the forged input image; forgery detection with false matches; the result of RANSAC; the masks that were generated (final result).

3.7 Neighbourhood Clustering

Christlein et.al [98] suggest Same Affine Transformation Selection (SATS) to estimate the affine transformation parameters of the copy-moved areas. To increase stability they iteratively refine their basic transformation estimation.

We found that there is no need to include this step and used RANSAC to remove the outliers. We propose a simple version of SATS to remove false matches by analysing the neighbourhood of a possible match. The copied and moved blocks each had to comprise at least three neighbouring blocks within a radius (r=32), see Figure 3-6.

We considered neighbourhood clustering to reduce the false matches and the number of outliers. Without neighbourhood clustering, RANSAC may fail to estimate the transformation accurately because of the large number of outliers in some images.

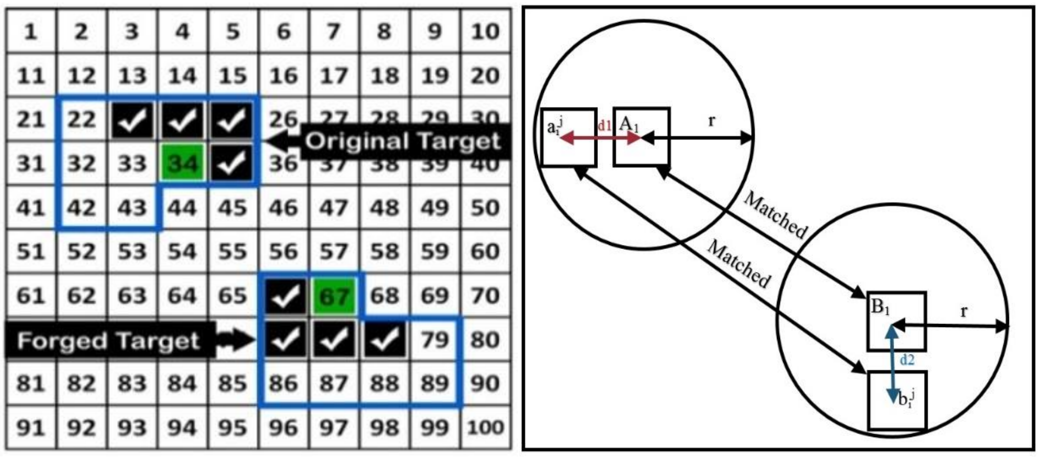

Figure 3-6: an example of matching two blocks using neighbourhood clustering; right: a block diagram of two matched blocks using neighbourhood clustering within a radius [99].

3.8 Hysteresis Technique

To produce the best possible result in the CMF detection, we used a hysteresis technique. Hysteresis thresholding is based on using two thresholds, one low and one high; it considers spatial information to improve the result. This technique has been employed in edge detection [100].

Recently, the hysteresis technique has been used in forgery detection [68]; the “strong” features have been detected using the low threshold, and the high threshold has been used to find the “weak” features. From our experiment, we found the performance of the automatic thresholding to be good, although it may overestimate the required threshold. Therefore, we need some post-processing technique to remove the false matching.

To use hysteresis thresholding in CMF detection, we used the OptimumThreshold, which is estimated using our proposed method in Section 3.4, and developed the following approach.

Find the 2ANN for each feature vector within the low threshold (OptimumThreshold/2) [100] [101] and this will reduce the number of false matches. The low threshold is used to detect similar “strong” features, which represent the pixels from the original and from the duplicated regions. RANSAC is applied to remove the outliers and find the transformation (coordinates) of the matched features.

For each coordinate in the transformation list, use the block dimensions to recolour the surrounding area. In the next step, dilate each region using a disk with a one-pixel radius. For each of the newly added pixels, use the original list of the feature vector to retrieve the corresponding information. Find the 2ANN for each new feature vector. If the Euclidean distance between the matched features is less than the high threshold (OptimumThreshold), store the coordinates of these features.

RANSAC is applied to remove any new outliers and keep the new coordinates within the previously found transformation. Add the new pixels to the matching list and update the transformation matrix. Grow the detection regions by adding a new block located at the centre of the new matched pixel. Repeat this process until no more pixels can be added to the primary detection. This region growing technique depends on the primary detection of the strong features matching the spatial information, to add one block each time to the edges of the primary detection [102].

The high level description of the proposed method to use hysteresis technique with CMF detection is as follows:

```
Input: List A, List B of coordinates, where $A_i$ match $B_i$ , The whole
Feature Vectors(FV) list.
Output: List C and List D, where $C_j$ match $D_j$.
    1. Use List A, List B and the block dimensions to build a binary
       mask.
    2. While Boolean ==true
        a) Dilate each region using a disk has a one-pixel radius.
        b) Use the coordinates of each newly added pixels to find
           the corresponding feature vector in the (FV) list.
        c) Build a k-d tree for the new feature vectors and find the
           2nd Approximate Nearest Neighbours for each element in
           the tree.
        d) Find the matched feature vectors (blocks) which satisfy:
            i.  The Euclidean distance between feature vectors is
                less than $\text{Threshold}_1 = (\text{OptimumThreshold})$.
            ii. The distance between the centres of the two
                matched blocks is less than $\text{Threshold}_2 = 16\sqrt{2}$.
        Save the coordinates of new matched blocks/features in
        List E, List F.
    e)  If (List E) neighbour to (List A)
            List C= concatenate (List A, List E)
            List D= concatenate (List B, List F)
        else
            List C= concatenate (List A, List F)
            List D= concatenate (List B, List E)
        end
    f)  Call RANSAC to estimate the transformation of the List C
        and List D.
    g)  If the size of (List A == List C and List B == List D)
            Boolean = false
        End
3.  End
```

Figure 3-7 shows an example of growing the detection regions which increases the F-measure from 0.89 to 0.93.

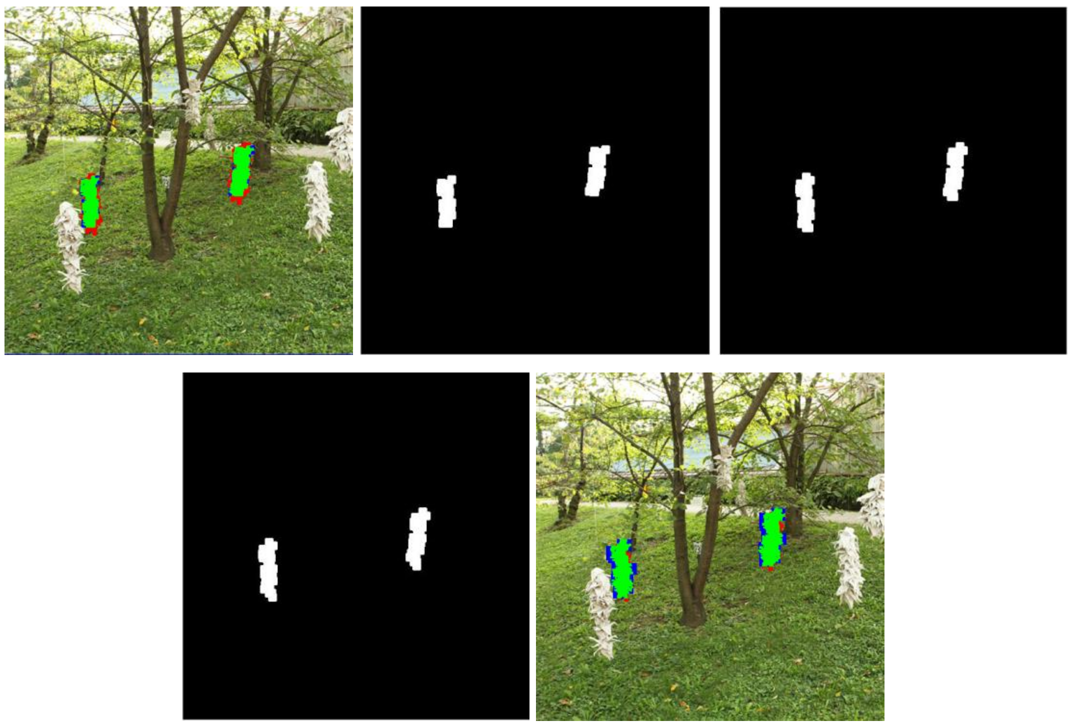

Figure 3-7: An example of growing the detection regions with hysteresis thresholding.

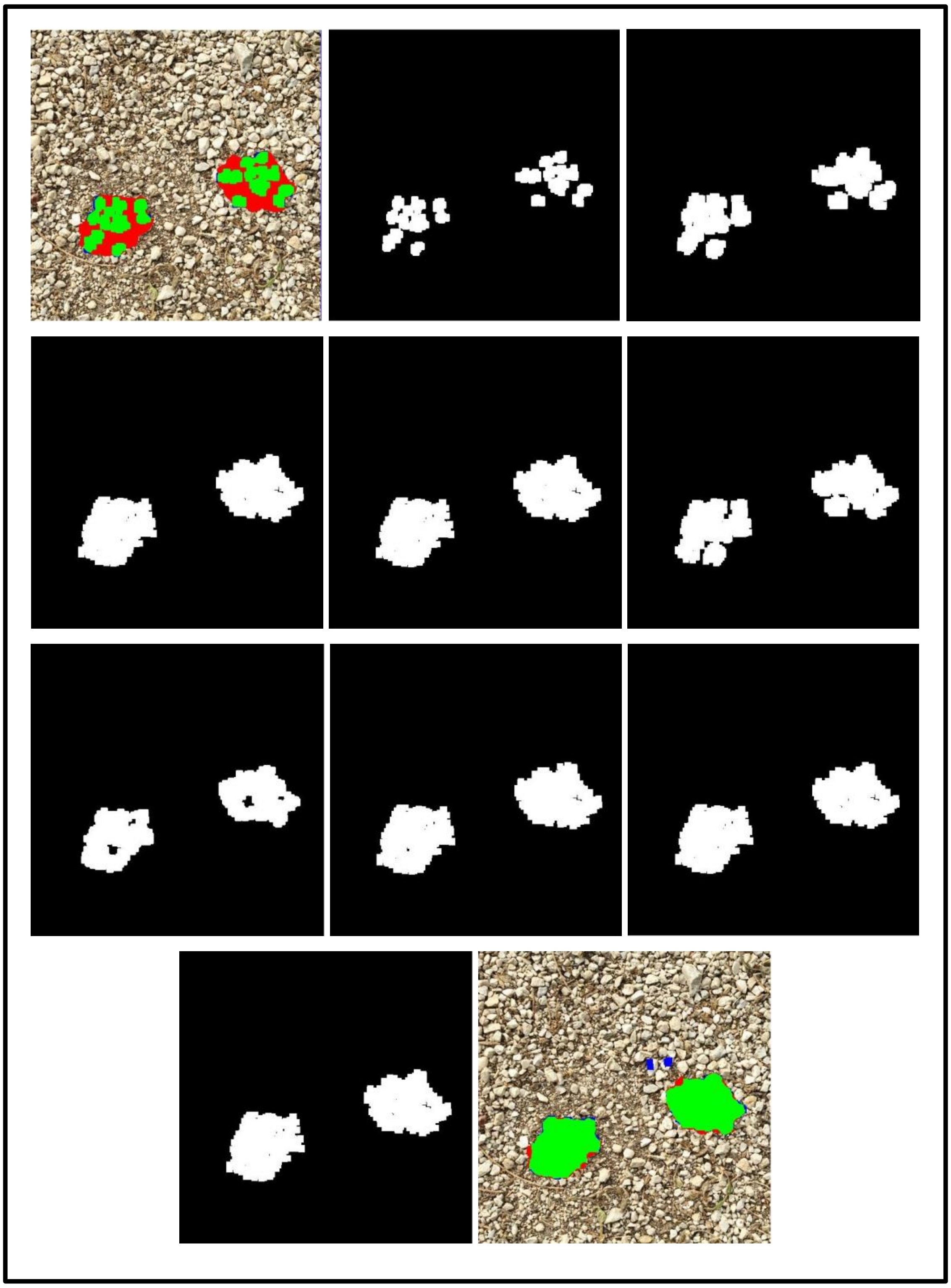

Figure 3-8: An example of growing the detection regions with hysteresis thresholding.

As illustrated in Figure 3-8, the detection regions were grown, and the F-measure increased from 0.69 to 0.97.

3.9 The Experiments

As previously explained in Section 2.3.1, we consider CoMoFoD dataset (small images) [2] in order to evaluate our algorithms.

3.9.1 The Difference between Matching Points Algorithms in CMF Detection

Previous works suggested two major methods for finding similar blocks in CMF Detection. The first method is sorting the feature vectors lexicographically and computing the similarity value between blocks (the Euclidean distance). The second method is building the k-d tree and finding the 2ANN. We tested both methods and found them to have similar effectiveness for translation CMFD, but the first method failed with rotation and we could not detect forgery with it.

To understand the reason for the failure of the first method, we carried out the following experiment: We computed the descriptors for two cloned blocks and saved them. Then we built the descriptors of the whole image, sorted them lexicographically and searched for the two saved descriptors. If lexicographic sorting worked properly with our method, the two saved descriptors would have been adjacent. We found that there were 189 descriptors between the two saved descriptors. The reason for this is that lexicographic sorting works like a dictionary. Consider that the interpolation generates changes between the descriptors of the original and the rotated patches. In that case, lexicographically sorting obviously cannot be used to detect forgery with rotation.

3.9.2 CMF detection with translation and Original Genuine Images

The CoMoFoD contains 40 different images with plan CMF, we tested all available images and could detect forgery in all images, but also incurred some false detections, see Figure 3-5. To remove the false matching, we tested three different methods; RANSAC produced the best results with a very short run time, as shown in Table 3-1.

Table 3-1 The results of experiments with translation.

Post-processing Method to Remove False Matching	**F-Measure**	**Total Running Time For 40 images**
Without Post-processing	0.87	155 sec
Shift Vector	0.87	45 min
Neighbourhood Clustering	0.91	210 sec
RANSAC	0.93	170 sec

Then we used the same pipeline with Zernike moments to test the same images. The improved SIFT produced similar or better results than the Zernike moments, see Figures 3-9, 3-10, 3-11 and 3-12.

We conducted an experiment on 40 different CoMoFoD original genuine images (without forgery), and the algorithm produced an excellent result as the F-measure = 0.994 and detected no forgery on 37 images out of 40.

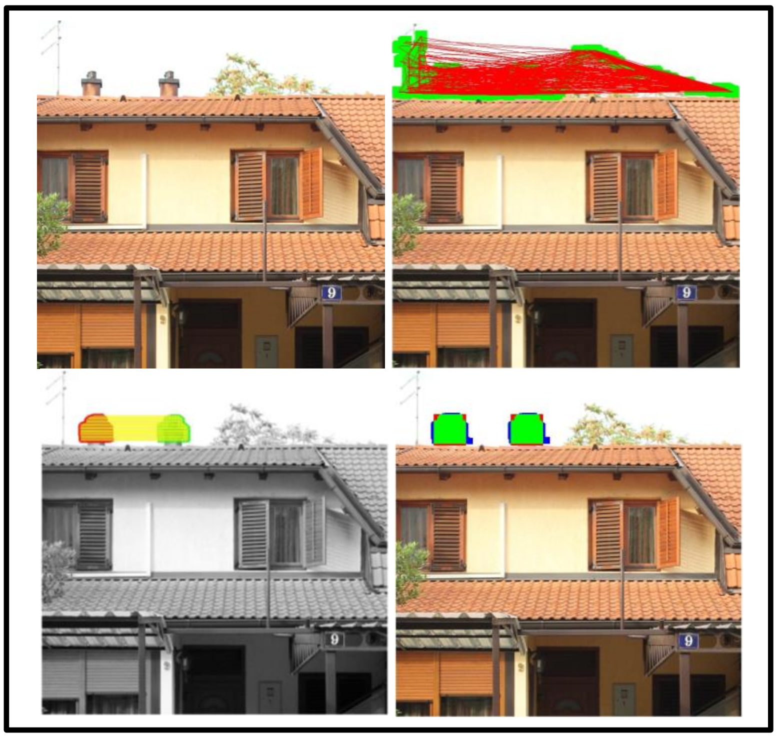

Figure 3-9: An example of improved DSIFT (left to right, top to bottom) input image A, primary detection RANSAC result, final result after hysteresis.

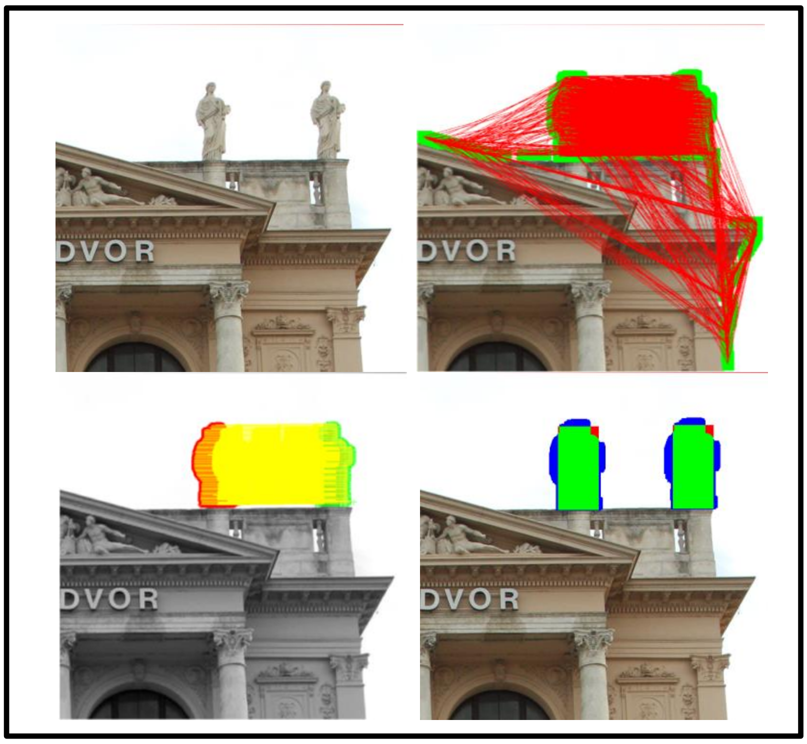

Figure 3-10: An example of Improved DSIFT (left to right, top to bottom) input image B, primary detection RANSAC result, final result after hysteresis.

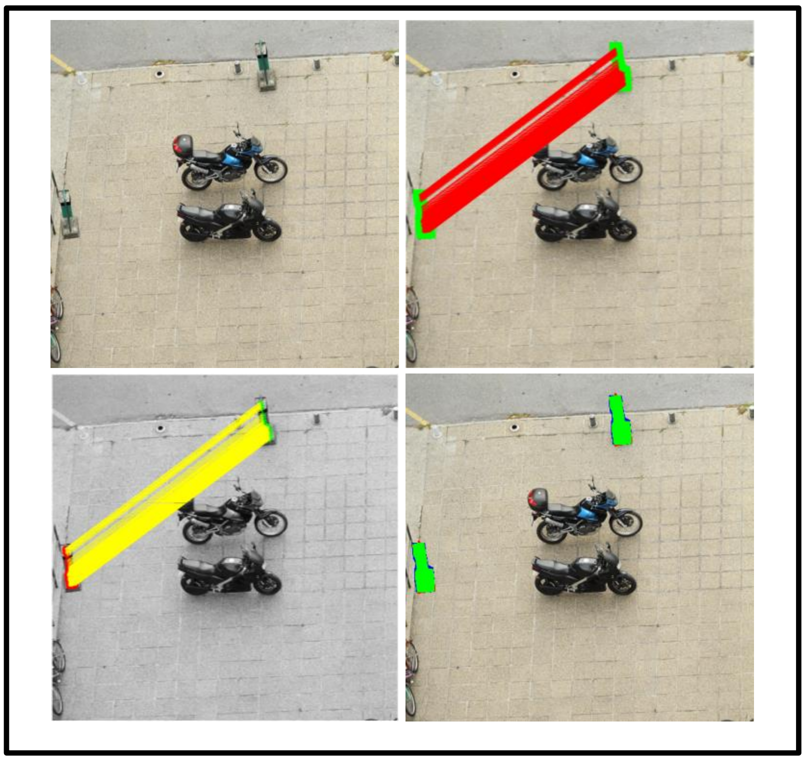

Figure 3-11: An example of improved DSIFT (left to right, top to bottom) input image C, primary detection RANSAC result, final result after hysteresis.

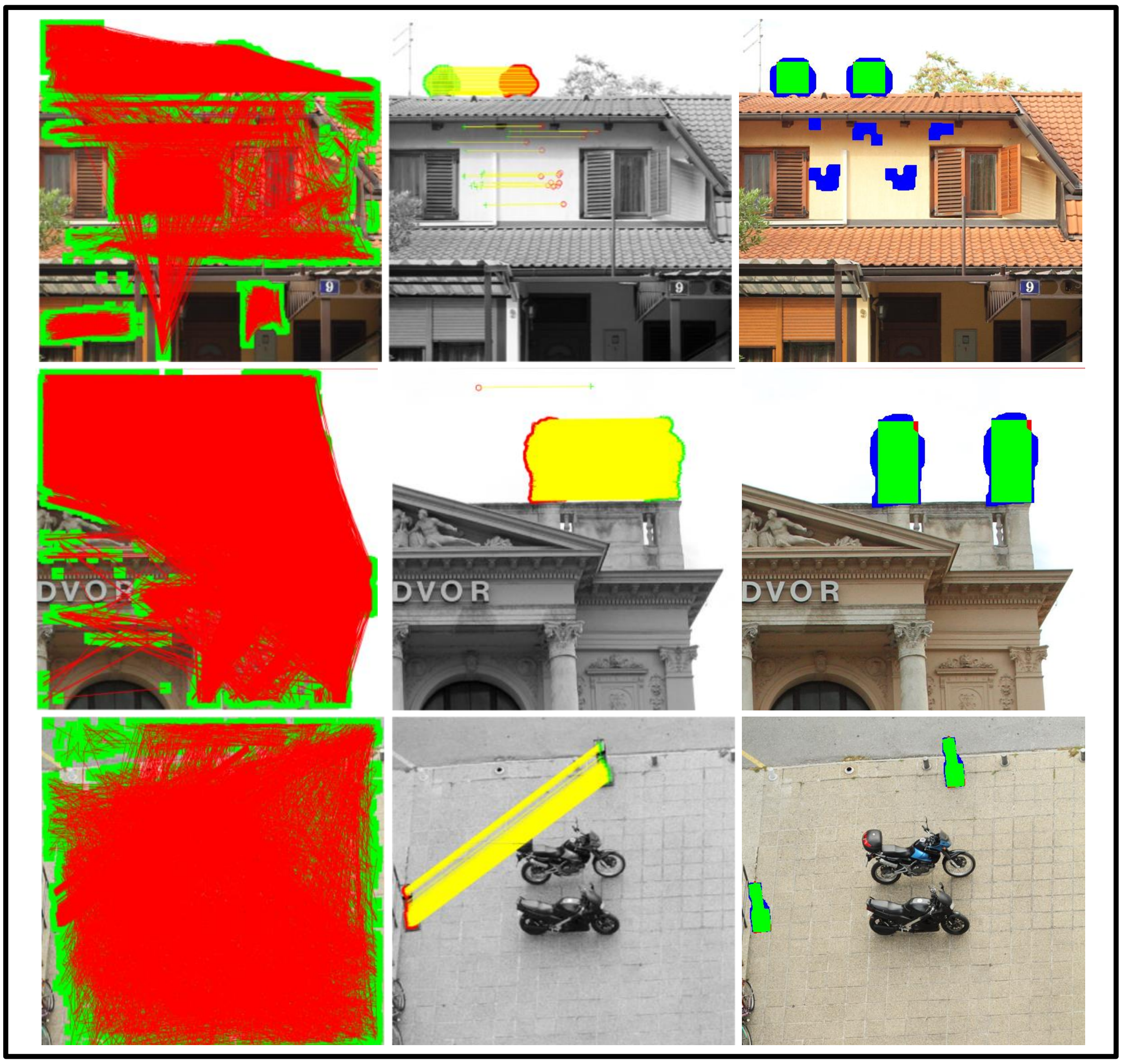
Figure 3-12: Using Zernike moment to detect forgery, (left to right) primary detection, RANSAC result, final result after hysteresis.

3.9.3 CMF detection with translation and post-processing (attacks)

To create more realistic CMF images and to hide the traces of forgery, the forger may use post-processing methods. In our work, we considered different types of attack (image blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise). We used our suggested method to test 200 images with different types of post-processing. Then we used the same pipeline with Zernike moments to test the same images. In most cases, our improved DSIFT produced better results than the Zernike moments did. Our method detected the forgery in 198 images out of 200, see Figure 3-13, Figure 3-14, Figure 3-15 and Figure 3-16.

Table 3-2 Comparison between improved DSIFT and Zernike moments for CMF detection with different types of post-processing (Avg. of F-Measure).

Post-processing	**Improved DSIFT**	**Detected images using improved DSIFT**	**Zernike moments**	**Detected images using Zernike moments**
Image Blurring, (5×5 average filter)	0.78	38	0.56	34
Brightness Change Range (0.01, 0.8)	0.87	40	0.45	33
Colour Reduction (32 intensity levels)	0.90	40	0.49	33
JPEG Compression (quality factor=40)	0.78	40	0.60	33
Contrast Adjustment Range (0.01,0.8)	0.90	40	0.48	32

We used our improved DSIFT to detect CMF with different levels of added noise. We also used Zernike moments to detect CMF with the same noisy images.

We achieved satisfactory results, and got better results than we did with Zernike moments, see Table 3-3.

Table 3-3 Comparison between improved DSIFT and Zernike moments on detection CMF with different levels of noise (Avg. of F-Measure).

Value of White Gaussian Noise (AWGN)	**Improved DSIFT**	**Detected images with improved DSIFT**	**Zernike moments**	**Detected images with Zernike moments**
0.001	0.68	36	0.63	36
0.005	0.54	32	0.51	31
0.01	0.48	32	0.41	26

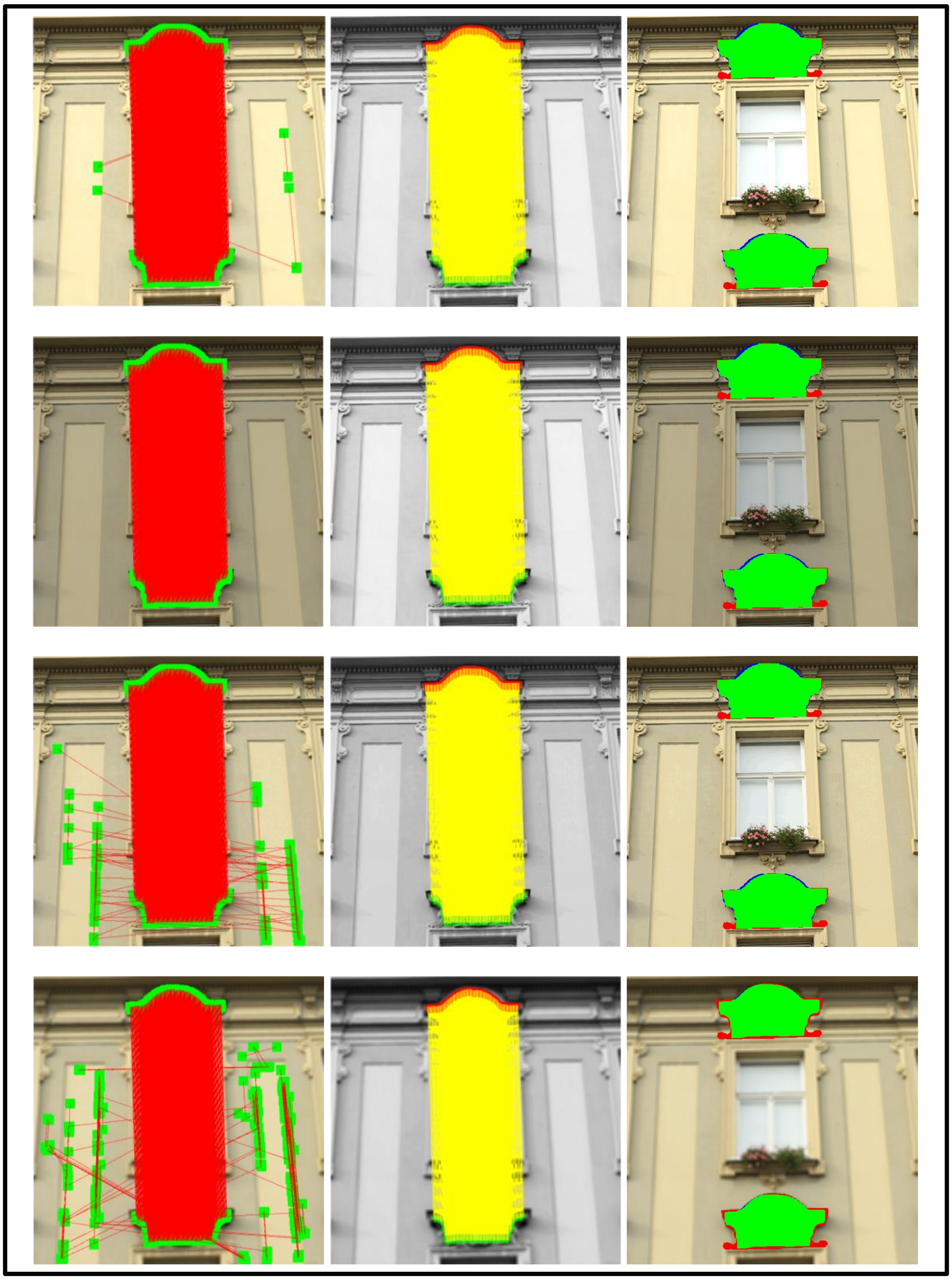

Figure 3-13: An example of using improved DSIFT to detect forgery in post processed images (1st row) the steps of detection in image with brightness change, (2nd row) the steps of detection in image with Contrast Adjustment, (3rd row) the steps of detection in image with Colour Reduction, (4th row) the steps of detection in image with Image Blurring.

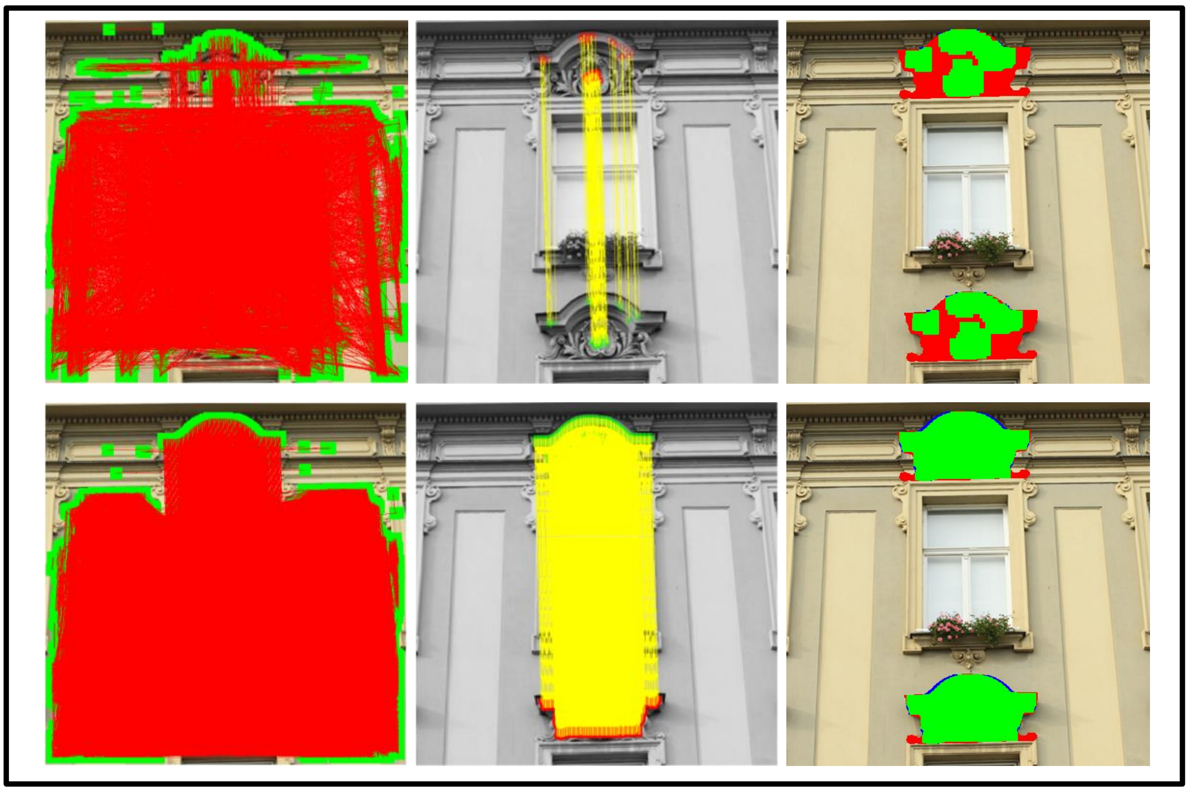

Figure 3-14: An example of using improved DSIFT to detect forgery in post processed images: 1st row, left to right, primary detection in a JPEG image; RANSAC result; final result after hysteresis. 2nd row, left to right, primary detection in noisy image; RANSAC result; final result after hysteresis.

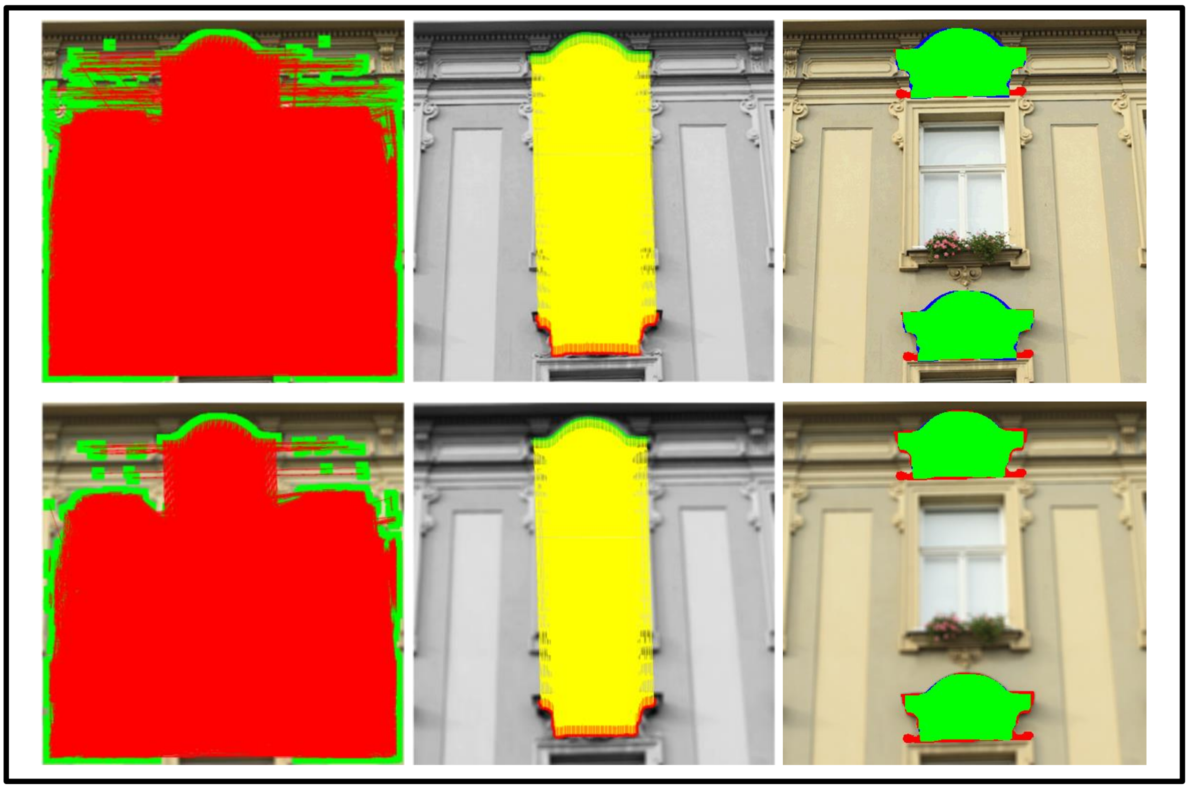

Figure 3-15: An example of using Zernike moments to detect forgery in post processed images: 1st row, the steps of detection in an image with Colour Reduction; 2nd row, the steps of detection in an image with Image Blurring.

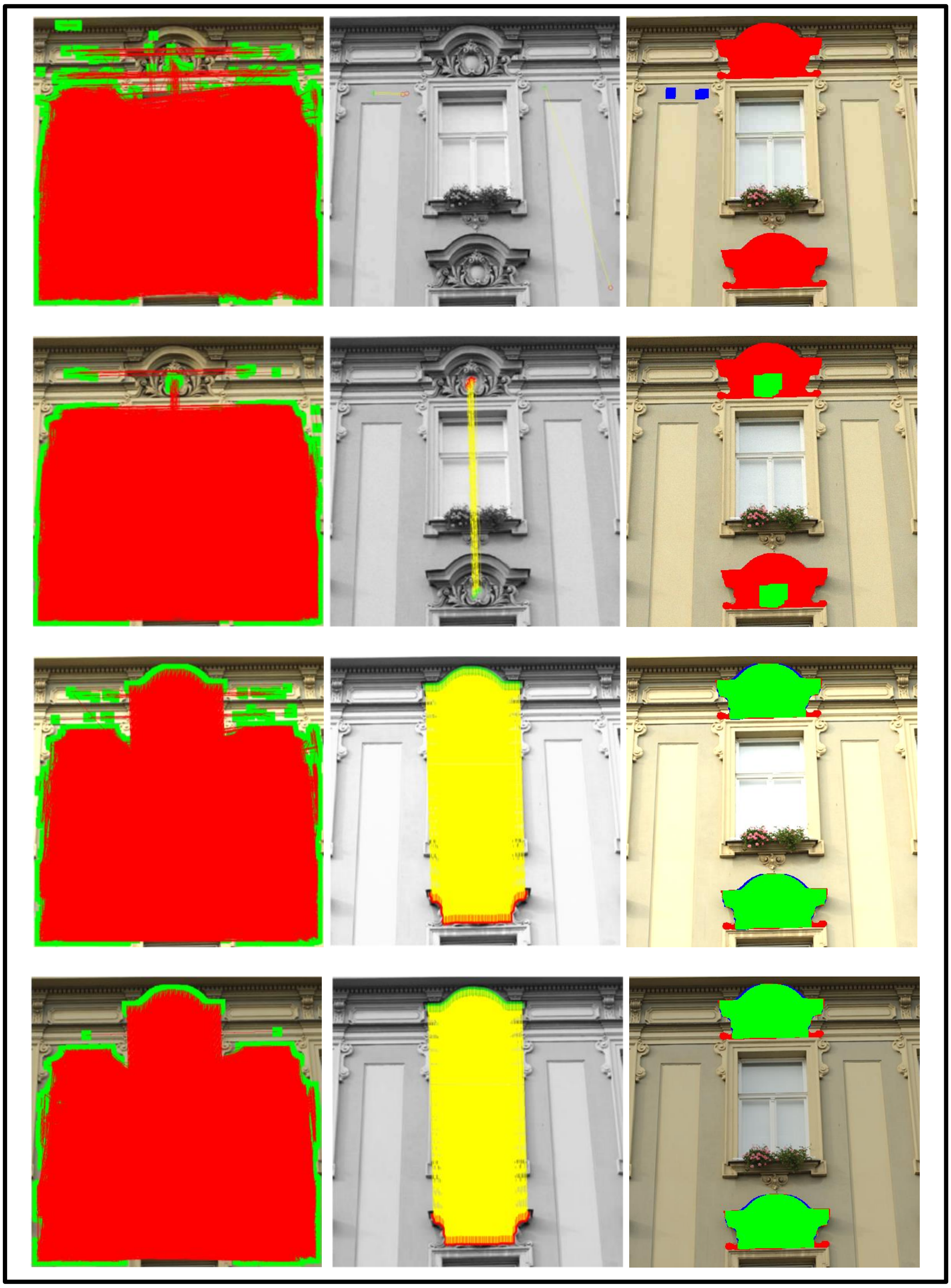

Figure 3-16: An example of using Zernike moments to detect forgery in post processed images: 1st row, the steps of detection in an image with JPEG compressed image; 2nd row, the steps of detection in an image with noisy image; 3rd row the steps of detection in an image with brightness change; 4th row) the steps of detection in an image with Contrast Adjustment.

3.9.4 Copy-Rotate-Move Forgery (CRMF) detection

An experiment was conducted on 40 different images with CRMF. These forged images had object(s) rotated by angles of different degrees (e.g. 180°, 90°, 10°, 2°, 4°, −4°, 5°, −7°, −3°, 1°...etc.). We successfully detected forgery on all the tested images where the F-measure was 0.85, see Table 3-4.

Table 3-4 Comparison between improved DSIFT, Zernike moments and RIDSIFT in 40 images with CRMF (Avg. of F-Measure)

Methods to remove false matches	**Improved DSIFT**	**Zernike moments**	**RIDSIFT**
Without post-processing	0.40	0.28	0.32
Fixed threshold, Postprocessing	0.76	0.68	0.54
Automatic thresholding, post-processing	0.78	0.71	0.55
Post-processing, fixed threshold & hysteresis	0.79	0.72	0.56
Post-processing, automatic thresholding & hysteresis	0.85	0.73	0.58

Table 3-4 shows the results of using our improved DSIFT, Zernike moments and RIDSIFT on 40 images with CRM objects. The table shows the effect of using the following:

1. The fixed threshold and automatic thresholding, see Section 3.4
2. The post-processing method to remove false matches, see Section 3.6.
3. The hysteresis technique to grow the initial matches, see Section 3.8

Below, some examples with different image properties are shown. Figure 3-17 shows the detection of multiple CRMF regions. Figure 3-18 and Figure 3-19 show images with homogeneous textures in which it is very hard to detect forgery with the naked eye. In Figure 3-20, the size of CMF region is very small and it is hard to detect duplicated regions.

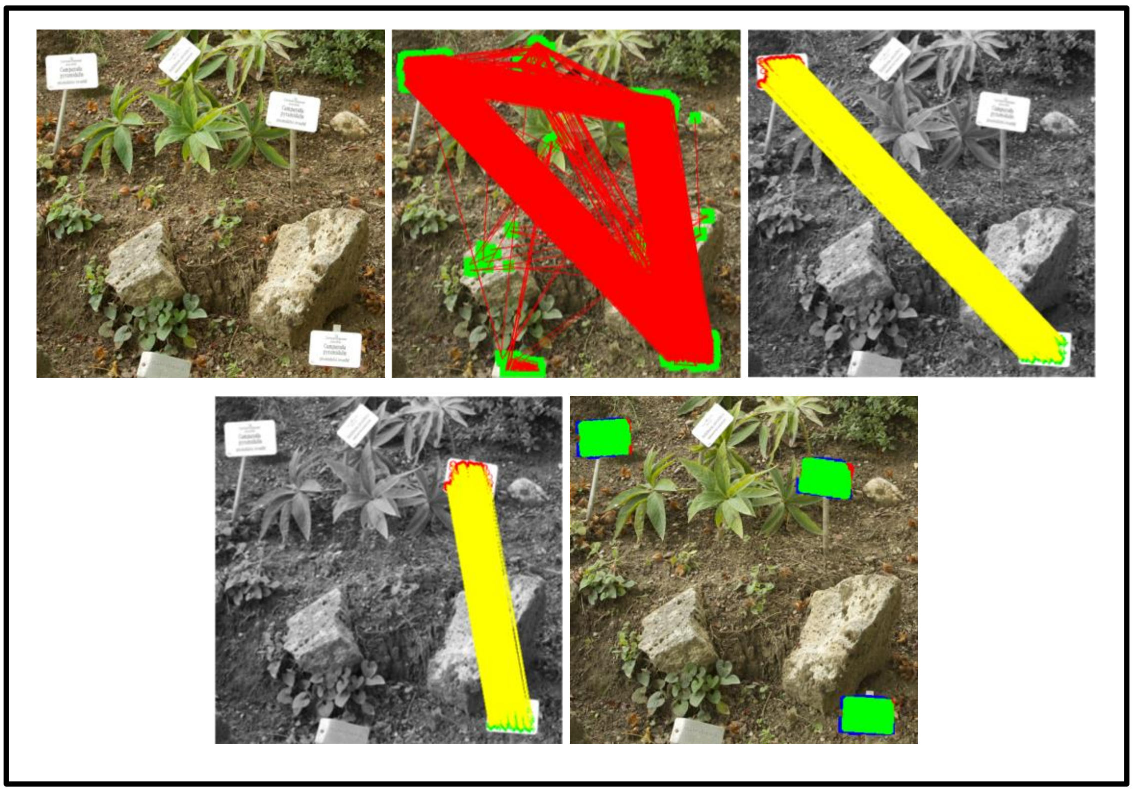

Figure 3-17: An example of detecting multiple duplicated objects using Improved DSIFT, (top to bottom, left to right) Forged image, copied object rotated by 5° and -7°, primary detection, RANSAC result, RANSAC result, final result after hysteresis.

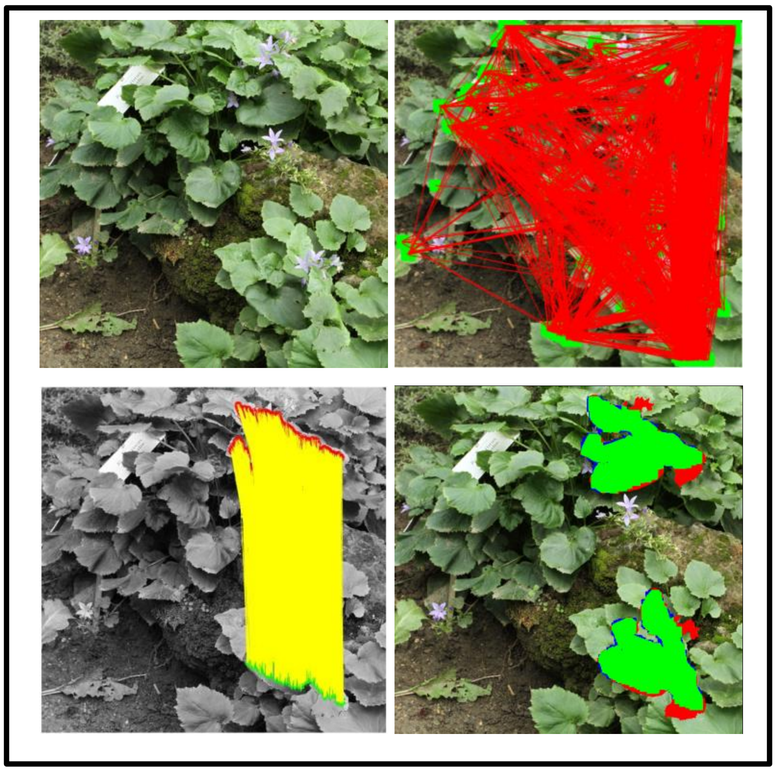

Figure 3-18: An example of detecting duplicated objects using improved DSIFT (top to bottom, left to right) Forged image, copied object rotated by 40°, primary detection, RANSAC result, final result after hysteresis.

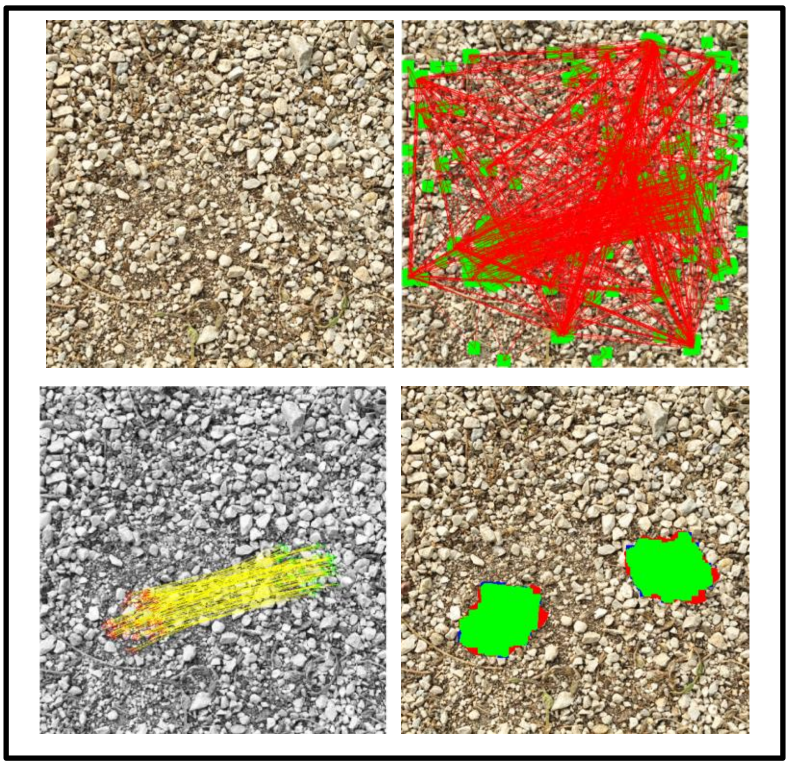

Figure 3-19: An example of detection duplicated objects using improved DSIFT, (top to bottom, left to right) Forged image, copied object rotated by 50°, primary detection, RANSAC result, final result after hysteresis.

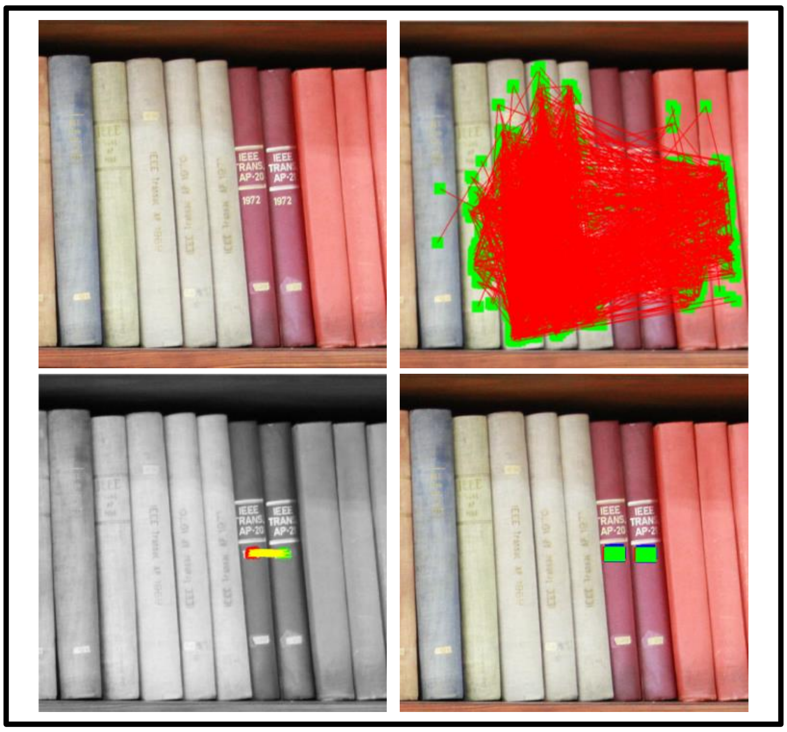

Figure 3-20: An example of detecting duplicated objects using improved DSIFT (top to bottom, left to right) Forged image, copied object rotated by -1°, primary detection, RANSAC result, final result after hysteresis.

As illustrated in Figure 3-21 and Figure 3-22, the Zernike moments are less efficient than improved DSIFT in forgery detection.

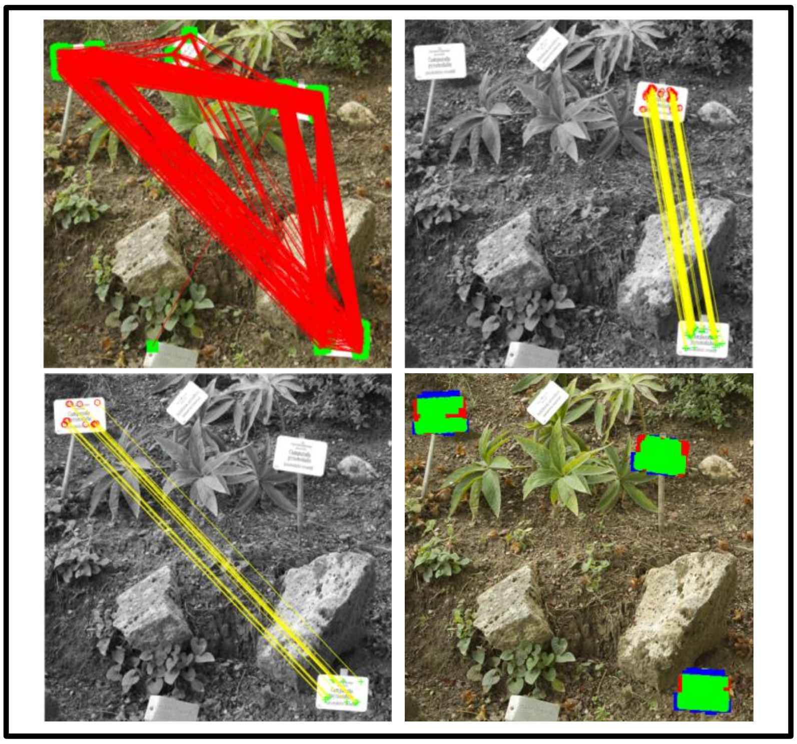

Figure 3-21: An example of detecting multiple duplicated objects using Zernike moments: (top to bottom, left to right) primary detection, RANSAC result, RANSAC result, final result after hysteresis.

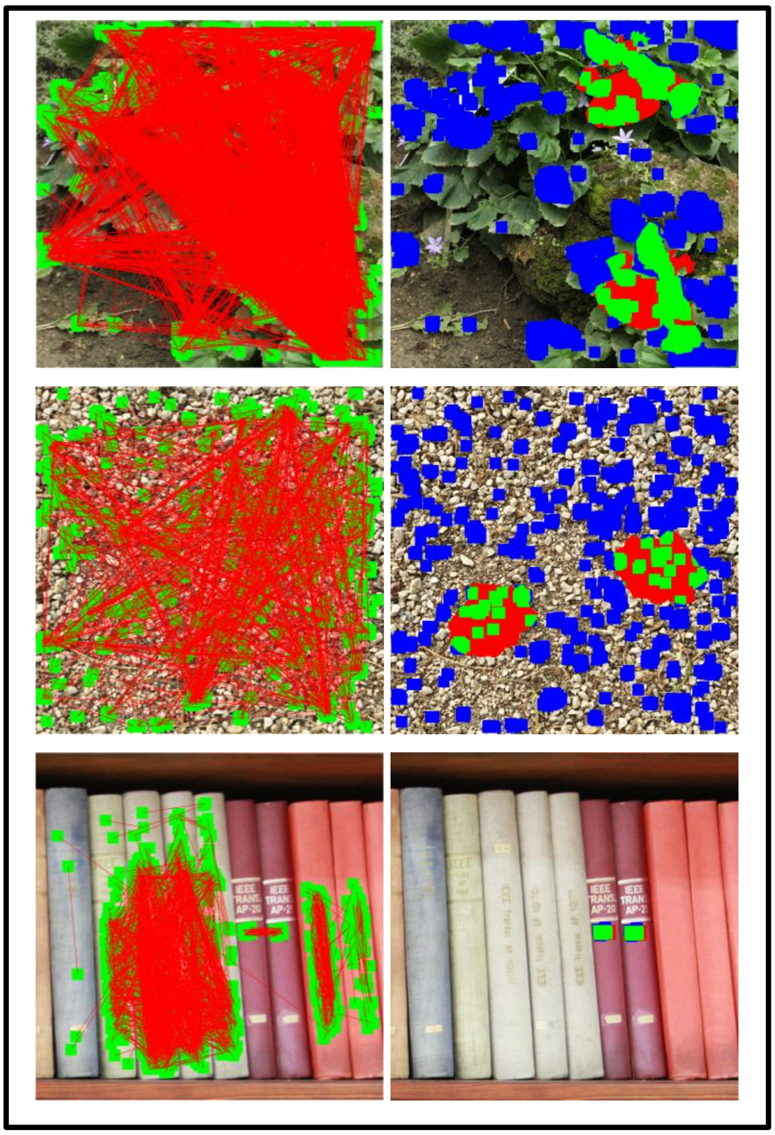

Figure 3-22: An example of detecting duplicated objects using Zernike moments: (left) Forged image, (right) final result after hysteresis.

Then, we considered CRMF detection with different types of post-processing (image blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise), the results are shown in Table 3-5.

Table 3-5 Using Improved DSIFT for CRMF detection with different types of post-processing (Avg. of F-Measure).

Post-processing	**F measure**	**Detected images using improved DSIFT**
Rotation	0.85	40
Image Blurring,(5x5 average filter)	0.67	36
Brightness Change Range (0.01, 0.8)	0.70	36
Colour Reduction (32 intensity levels)	0.70	36
Contrast Adjustment Range (0.01,0.8)	0.70	36
White Gaussian Noise (AWGN) =0.001	0.65	31
JPEG (40%)	0.65	31

3.9.5 Copy-Move Forgery with Scaling

Since the standard SIFT [3] determines the characteristic scale of an interest point, it is invariant to scale (as well as rotation). Our improved DSIFT is also rotation invariant, but not scale invariant; yet it is robust to small changes in scale.

We conducted an experiment to test the robustness to scale of our improved DSIFT. The overall F-measure is 0.43 for 40 different images with different scale factors (40%-150%). Moreover, the improved DSIFT with the proposed algorithm detected forgery on 30 images out of 40 with various scaled duplicated regions, see Figure 3-23.

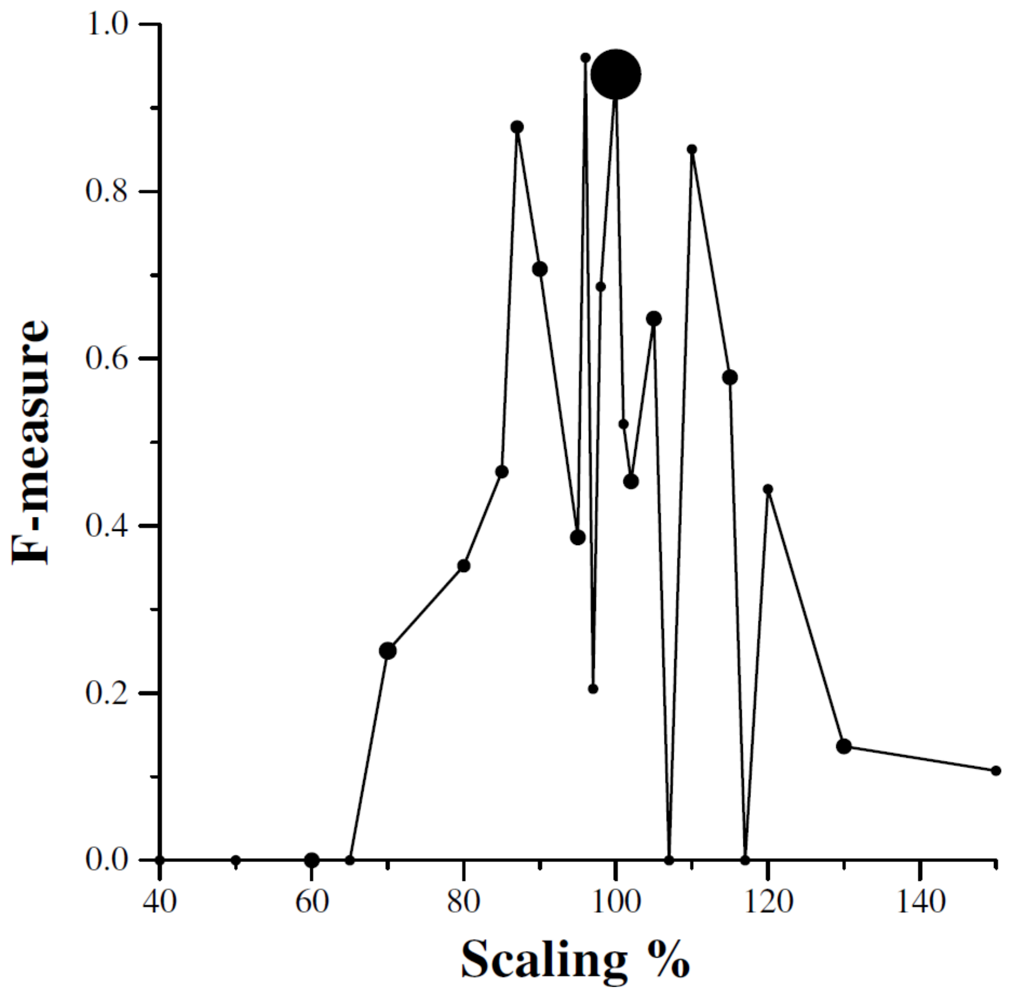

Figure 3-23: Average F-measure for 80 images with different scaling factors and the radius of each disk represents the square root of the number of images.

Here the radius of each disk represents the square root of the number of images. It can be seen that the graph is noisy, since the dataset [23] provides only a small number of test images for each scale factor. The low F-measure values with scaling factors close to 100% are generated from a small number of samples, as indicated by the small disks. Therefore, the descriptor/algorithm succeeds in detecting forgery on many different images with moderate scale factors. Below are some examples of detecting duplicated regions with different scales, see Figure 3-24, Figure 3-25 and Figure 3-26. Figure 3-27 shows an example of the failure of detection duplicated scaled objects where the CMF object has been scaled up by 50%.

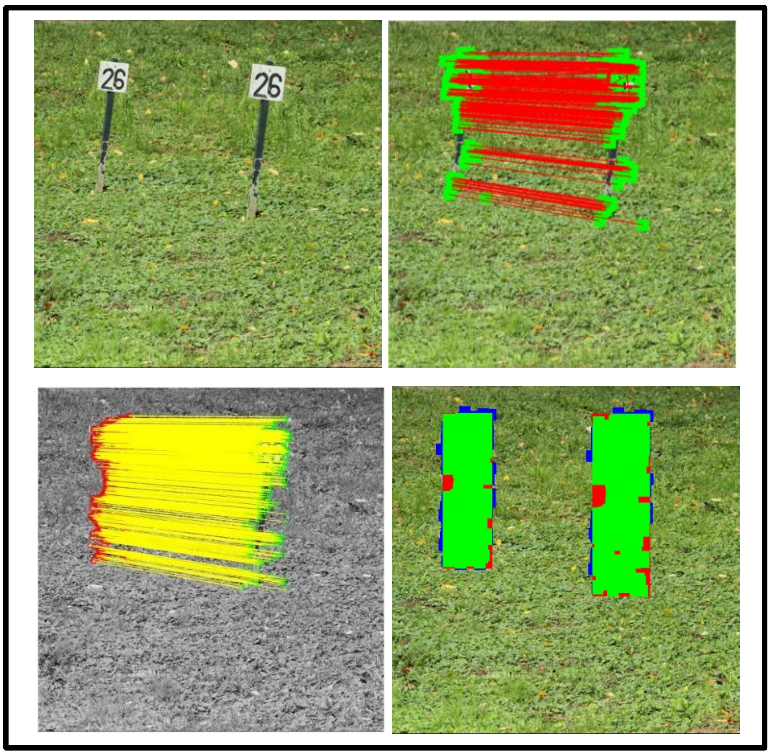

Figure 3-24 example of detecting duplicated scaled objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object scaled down by 15%, primary detection, RANSAC result, final result after hysteresis.

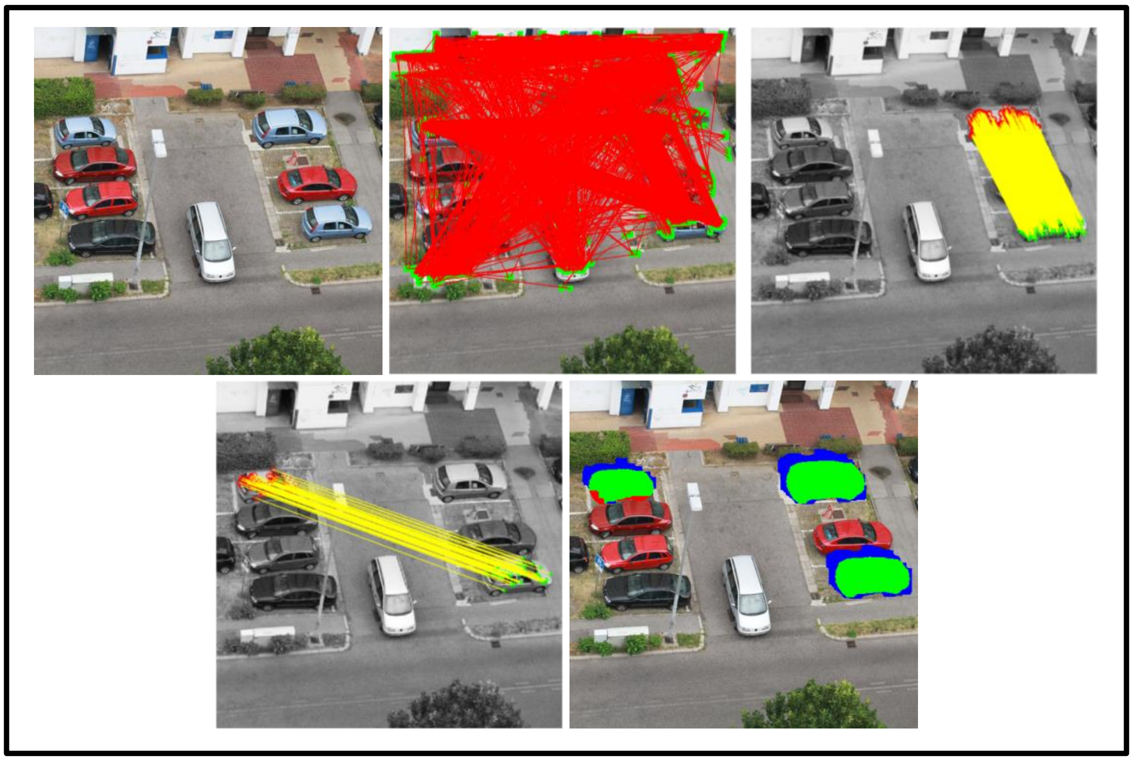

Figure 3-25 An example of detecting multiple duplicated objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object scaled down by 13% and scaled up by 5%, primary detection, RANSAC result RANSAC result, final result after hysteresis.

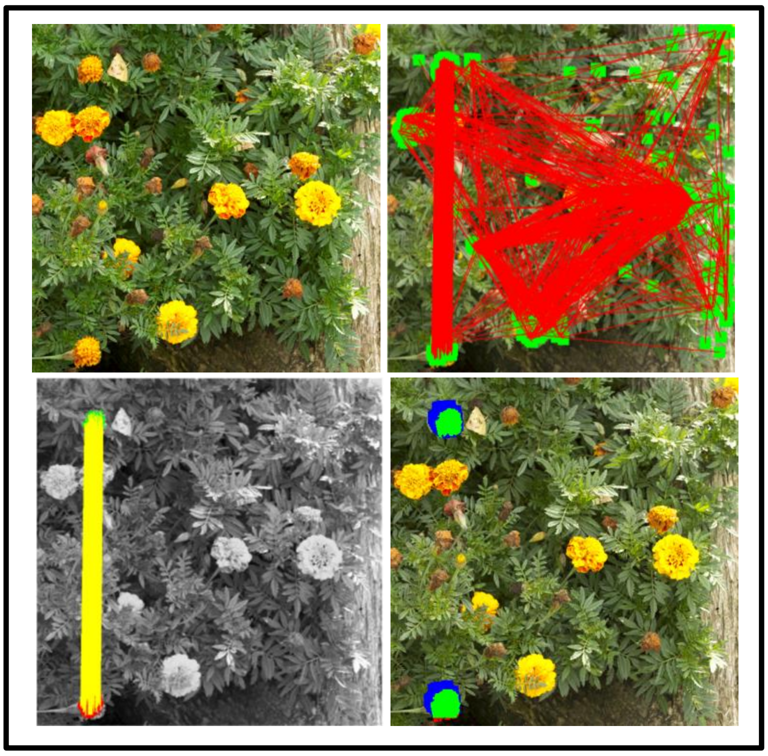

Figure 3-26: An example of detecting duplicated scaled objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object scaled up by 10%, primary detection, RANSAC result, final result after hysteresis.

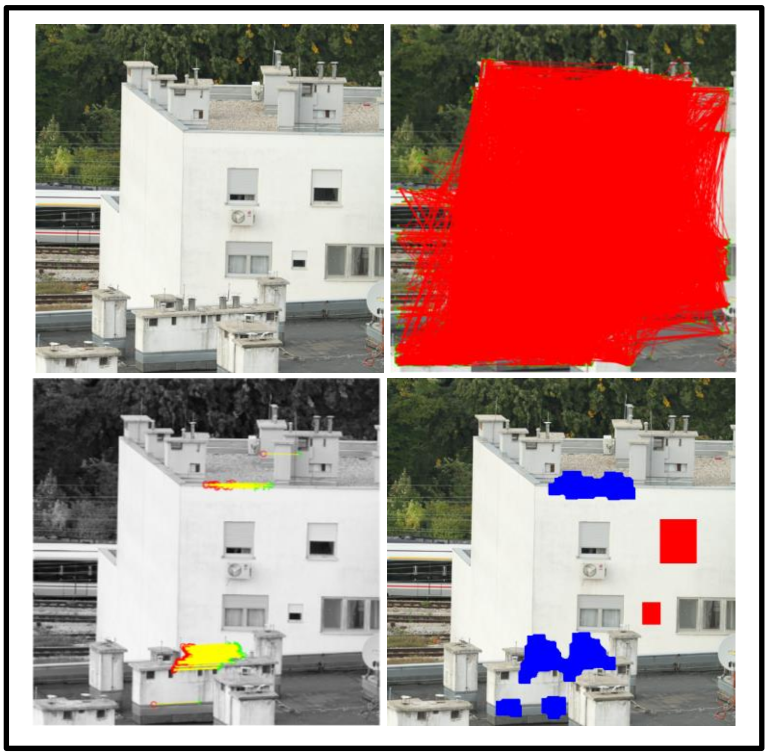

Figure 3-27 An example of failure of detecting duplicated scaled objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object scaled up by 50%, primary detection, RANSAC result, final result after hysteresis.

3.9.6 Copy-Move Forgery with Distortion

There are many types of image distortion (e.g. radial distortion, projection distortion, scale error, etc.) The dataset CoMoFoD contains CMF images with four different types of distortion (vertical elongation, horizontal elongation, vertical skew and horizontal skew).

We conducted an experiment to test the robustness of the improved DSIFT to distortion, and achieved good results. The overall F-measure is 0.60 for 40 different images with different distortion factors. Moreover, the improved DSIFT with the proposed algorithm detected forgery on 36 images out of 40 with various distorted duplicated regions, see

Figure 3-28, Figure 3-29 and Figure 3-30. Figure 3-31 shows an example of the failure of detection duplicated scaled objects where the CMF object has been horizontally elongated by 10%. The proposed method can detect forgery but it cannot remove the false matches, and we considered this result as failure of detection. Table 3-6 shows the details of the detection with more information on each type of distortion.

Table 3-6 the F-measure for each type of distortion.

Distortion type	**F-measure**	**Number of detected images out of 10**
Vertical scale (80%- 125%)	0.52	9
Horizontal scale (80%-115%)	0.50	8
Vertical skew (1%-8%)	0.73	10
Horizontal skew (1%-7%)	0.65	9

Tralic et al. [23] used the DCT to test 5 groups of images with 40 images in each group. They showed only the results where the F-measure value was greater than 0.5, that is, 2 images with scaling, 5 images with distortion, 3 images with the combination of these, 40 images with translation and 40 images with rotation. We do not compare our results with theirs, because we consider all the images in our test.

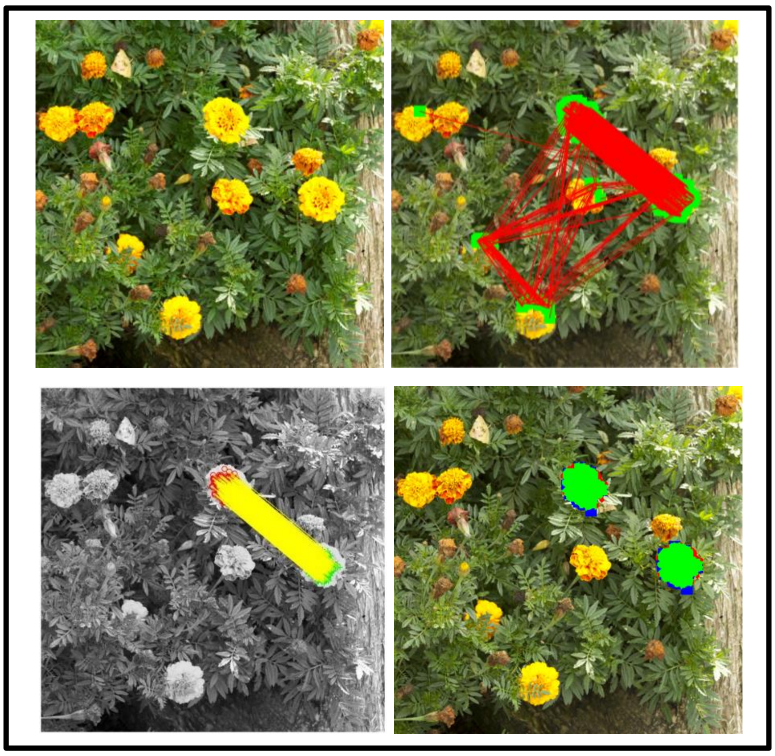
Figure 3-28: An example of detecting duplicated scaled objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object with a vertical skew factor of 8%, primary detection, RANSAC result, final result after hysteresis.

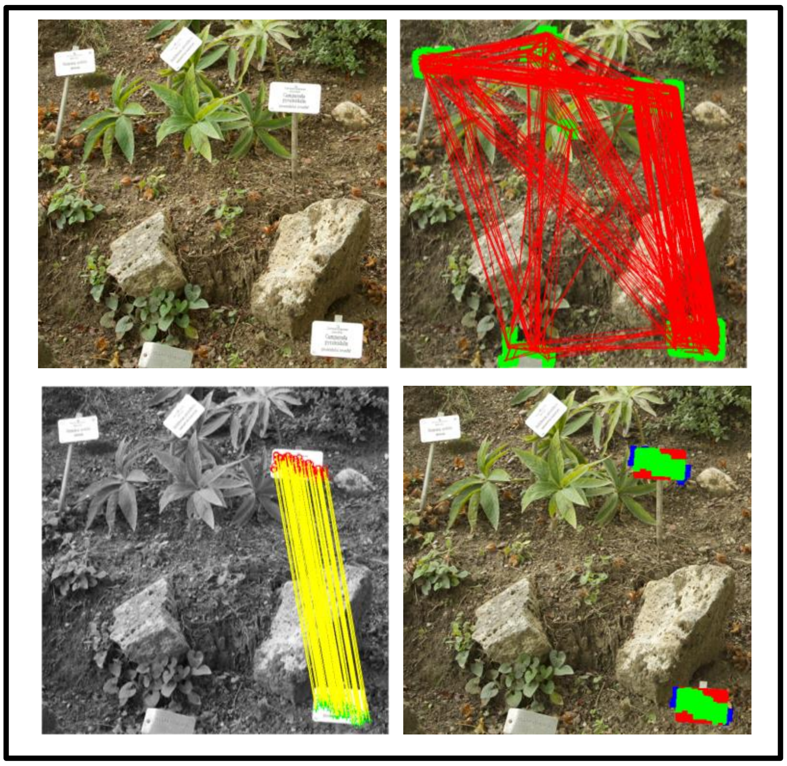
Figure 3-29: An example of detecting duplicated scaled objects using Improved DSIFT: (top to bottom, left to right) Forged image, copied object with vertical elongation by 15%, primary detection, RANSAC result, final result after hysteresis.

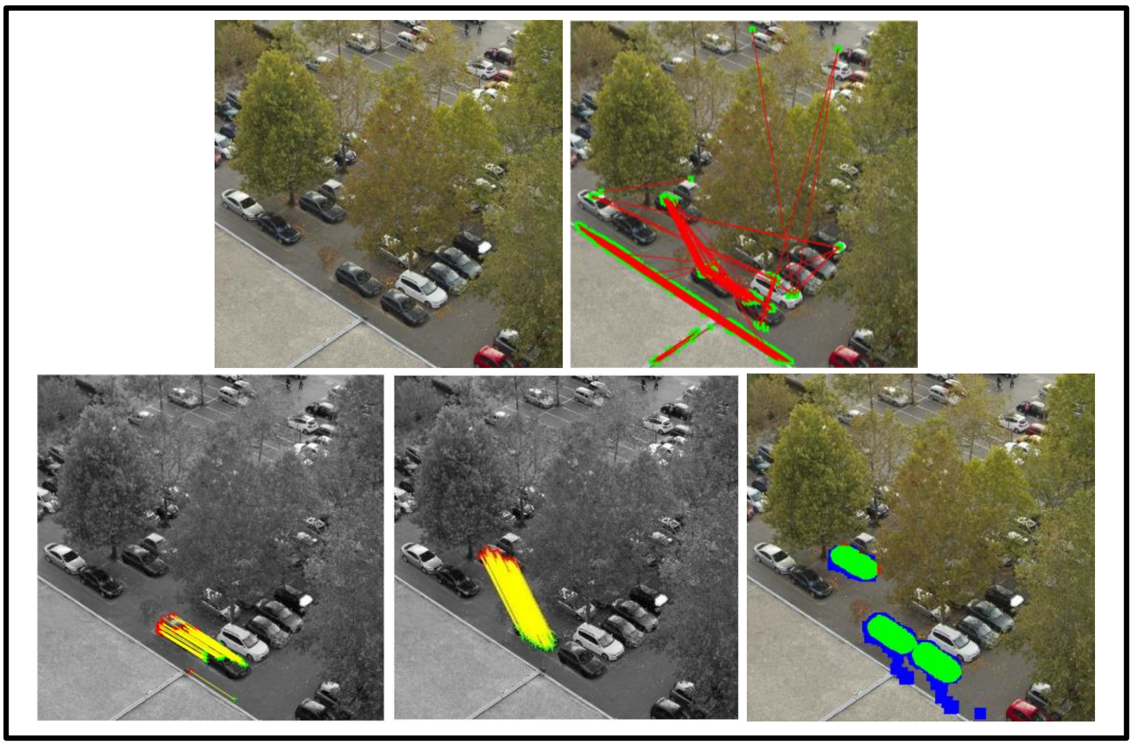

Figure 3-30: An example of detecting multiple duplicated objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object vertically elongated by 5% and by -10%, primary detection, RANSAC result, RANSAC result, final result after hysteresis.

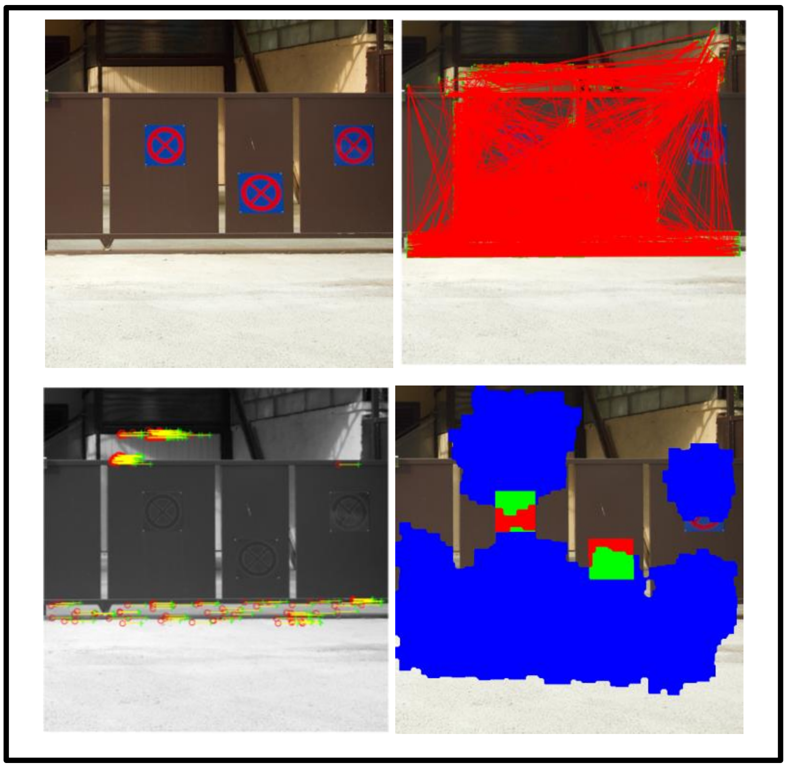

Figure 3-31 An example of failure of detecting duplicated objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object horizontally elongated by 10%, primary detection, RANSAC result, final result after hysteresis.

3.9.7 Copy-Move Forgery with Combined Transformation

This is the most challenging set of forged images because it shows two or more transformations being applied to copied region(s) before moving to a new location (e.g. the images have been scaled and rotated with different scaling factors or the images skewed horizontally and enlarged vertically). An experiment was conducted to test the robustness of the improved DSIFT/our algorithm to the combined transformation, and we achieved good results. The overall F-measure is 0.57 for 40 different images with different combination factors. Moreover, the improved DSIFT with the proposed algorithm detected forgery on 34 images out of 40 with various combined transformed duplicated regions, see Figure 3-32, Figure 3-33 Figure 3-34 and Figure 3-35.

Here, we cannot compare our results with the others because no equivalent results using the same dataset have been published.

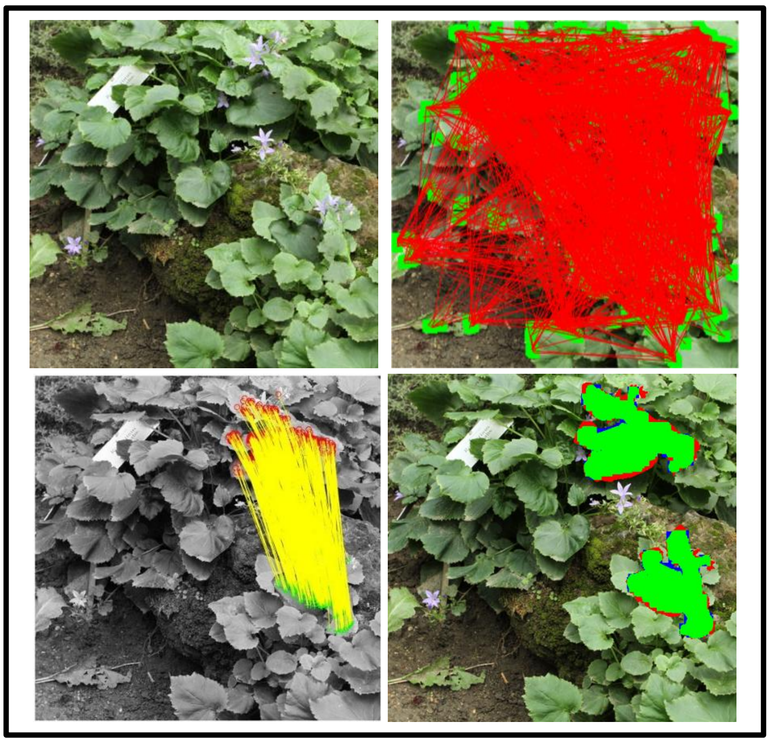

Figure 3-32: An example of detecting duplicated scaled objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object scaled down by 15% and rotated by 45°, primary detection, RANSAC, final result after hysteresis.

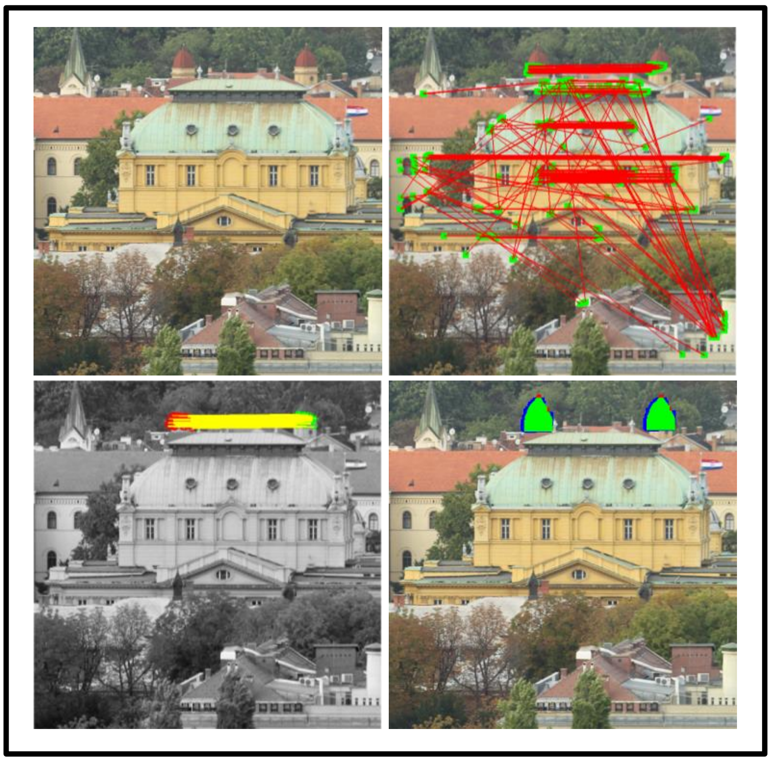

Figure 3-33: An example of detecting duplicated scaled objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object scaled up by 5% and vertically skewed by 1%, primary detection, RANSAC result, final result after hysteresis.

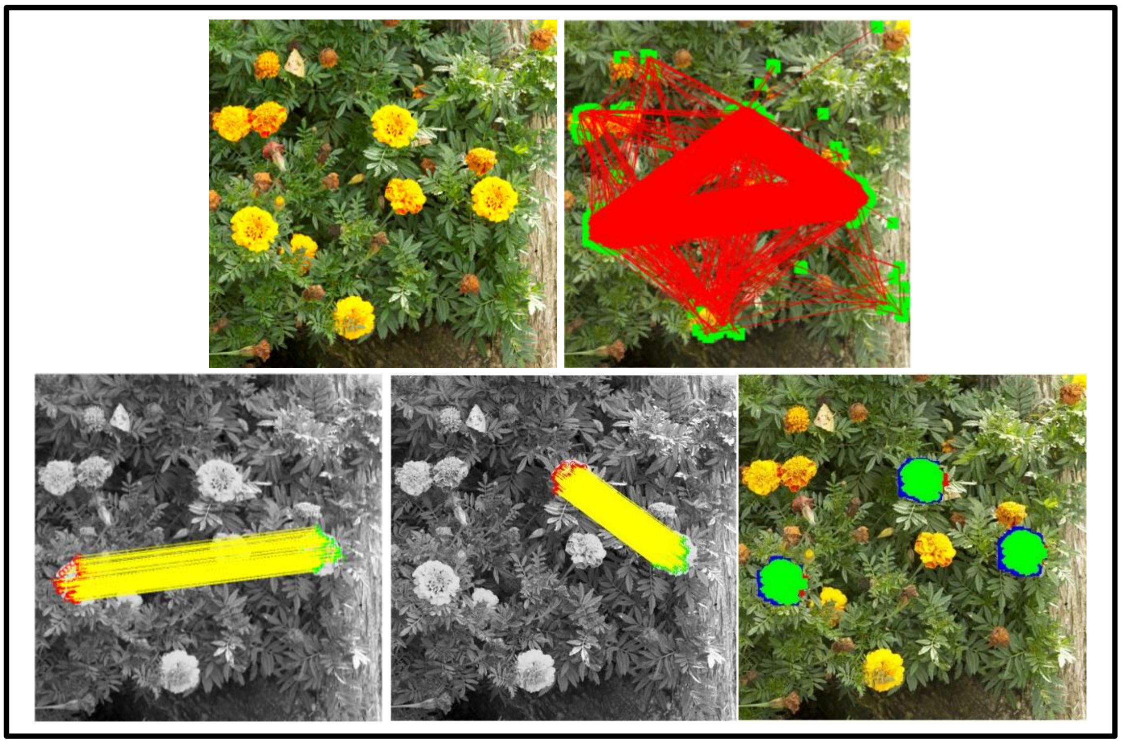

Figure 3-34: An example of detecting multiple duplicated objects using improved DSIFT: (top to bottom, left to right) Forged image, copied objects (horizontally elongated by 3% and vertically skewed by 2%, vertically elongation by -5% and horizontally skewed by 3%), primary detection, RANSAC result, RANSAC result, final result after hysteresis.

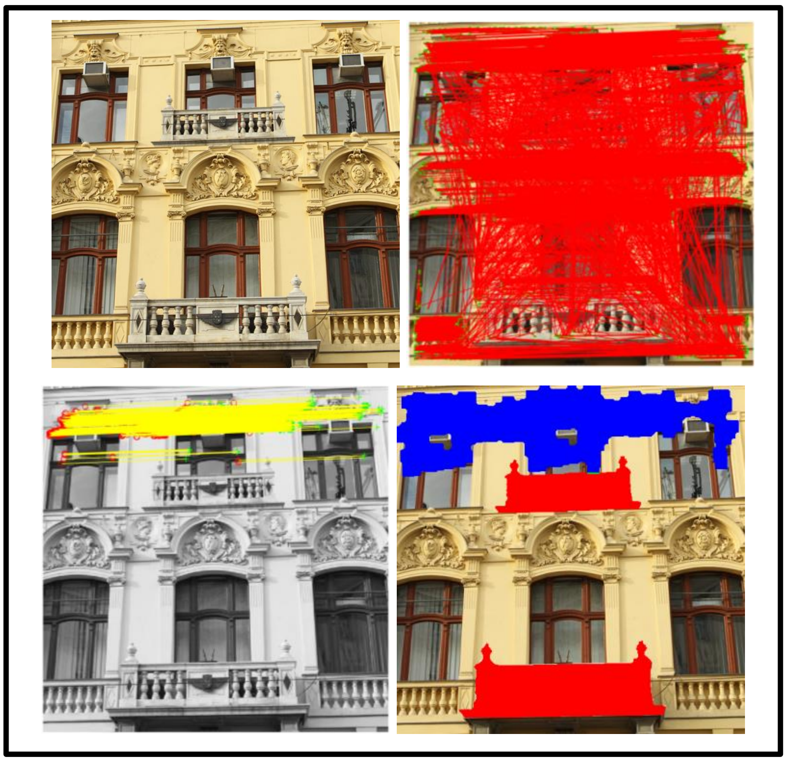

Figure 3-35 An example of failure of detecting duplicated scaled objects using improved DSIFT: (top to bottom, left to right) Forged image, copied object scaled up by 70% and rotated by -1%, primary detection, RANSAC result, final result after hysteresis.

3.10 Analysis of the proposed algorithm

To understand the improved DSIFT, we need to explain the differences between the standard SIFT and improved DSIFT. David Lowe has developed a Scale-Invariant Feature Transform algorithm [103] to detect and describe local features in images. The SIFT descriptor is invariant to translation, rotation and scaling transformations in the image domain, and it is robust to moderate perspective transformations and illumination variations [104]. SIFT keypoints (local extremas) are distributed sparsely on the image, and their location depends on the characteristics of the image itself.

The SIFT works very well with object recognition applications (e.g. Panorama stitching [105], 3D scene modelling[106], human action recognition [107], etc.).

Many papers use SIFT to detect copy-move forgery (e.g.[26] [56] [57] [19][70]); in general, these papers follow the same approach. They detect SIFT keypoints on the image, find the matched keypoints then clustering all spatially neighboured pixels in one group. Consider that, the standard evaluation method in the copy-move forgery detection is the F-measure on pixel level, as it is not enough to detect part of the duplicated objects. Although this approach worked well and can detect duplicated objects or part of it successfully, it has three main drawbacks which have a significant influence on its reliability:

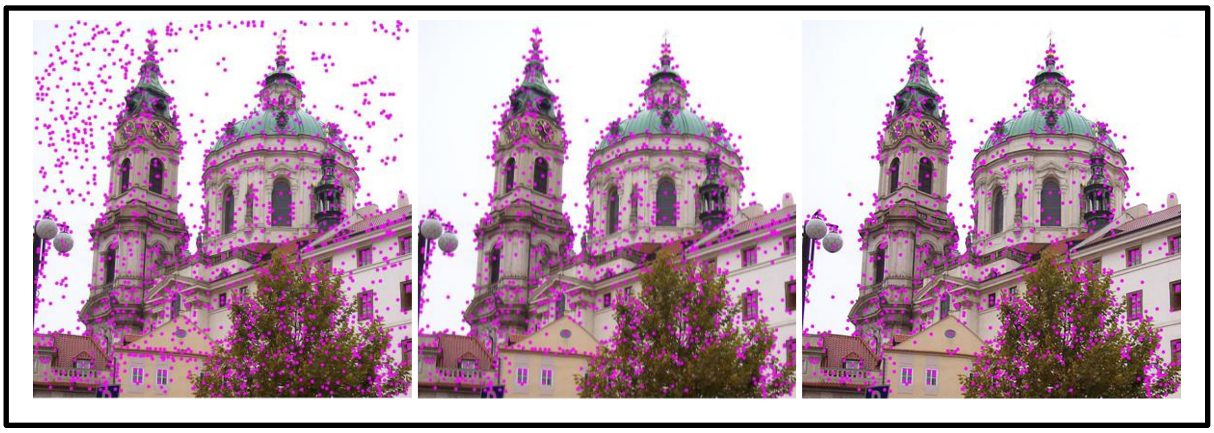

Figure 3-36: SIFT Keypoint Localization process, (left to right) the potential keypoints which have been detected by local extrema, eliminate low-contrast keypoints, eliminate edge keypoints and final result.

1. **SIFT rejects keypoints in flat regions:**

Flat regions occur where the pixel intensity values are similar to each other and change smoothly over comparatively large regions (e.g. sky, sea, sand, grass, etc.)[108]. The locations of potential keypoints, which have found by local extrema, refine in the SIFT Keypoint Localization step. As previously explained in section 3.2.1, the SIFT Keypoint Localization step eliminates any low-contrast keypoints and edge keypoints and remains the strong interest points, see Figure 3-36.

Since the SIFT reject keypoints in flat regions thus mean it fails to detect object forgery on such regions. Although using keypoints matching based approach (SIFT) can guarantee geometric invariance, but there is no keypoints on flat regions, which means it may fail to detect duplicated objects. As shown in Figure 3-37, where the black boxes are the copy-moved forgery regions, SIFT cannot detect keypoints on the flat regions (the wall).

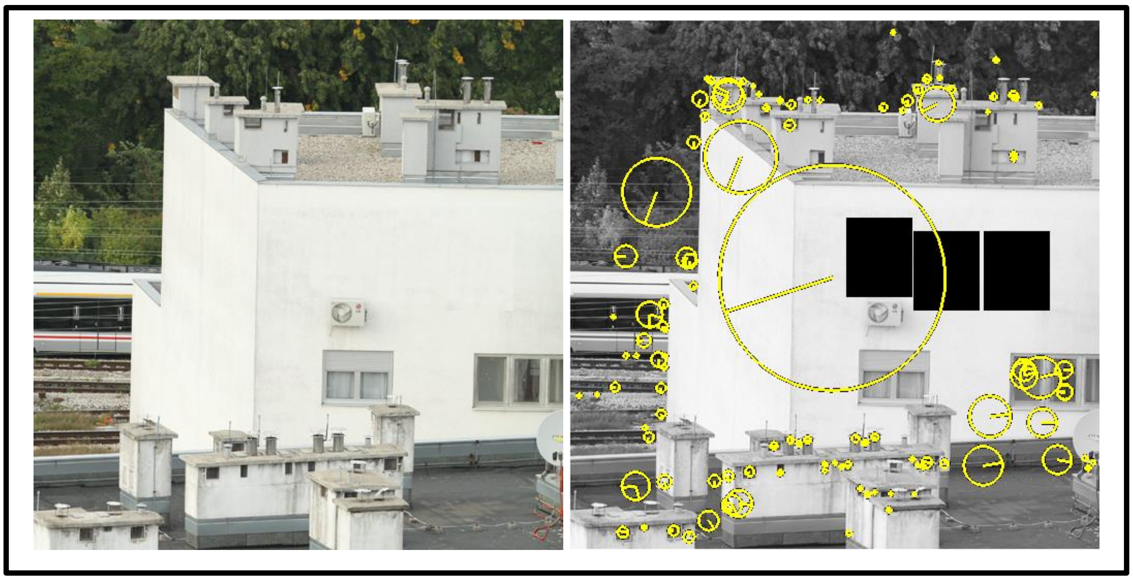

Figure 3-37: Using standard SIFT to find keypoints on image with flat region(s). (left) The input image (forged image), (right) SIFT keypoints with refinement step.

We performed an experiment to understand the effect on CMF detection of rejecting the unstable keypoints in SIFT, located in the low-contrast area(s) and edge(s).

We used SIFT in a different way to detect keypoints on the images with a flat region. All the detected local extrema points were used and the keypoint localization step was ignored. In other words, we kept the unstable keypoints on the flat regions. In such regions, the SIFT can detect some unstable keypoints, as shown on the top, right-hand image of Figure 3-38. Then we gradually increased the threshold to find the matched keypoints. But even when we increased the matching threshold to 100 times that of the initial threshold, it could find no match with other duplicated unstable keypoints. Moreover, no clustering technique has so far been proposed that can segment these flat CMF areas successfully.

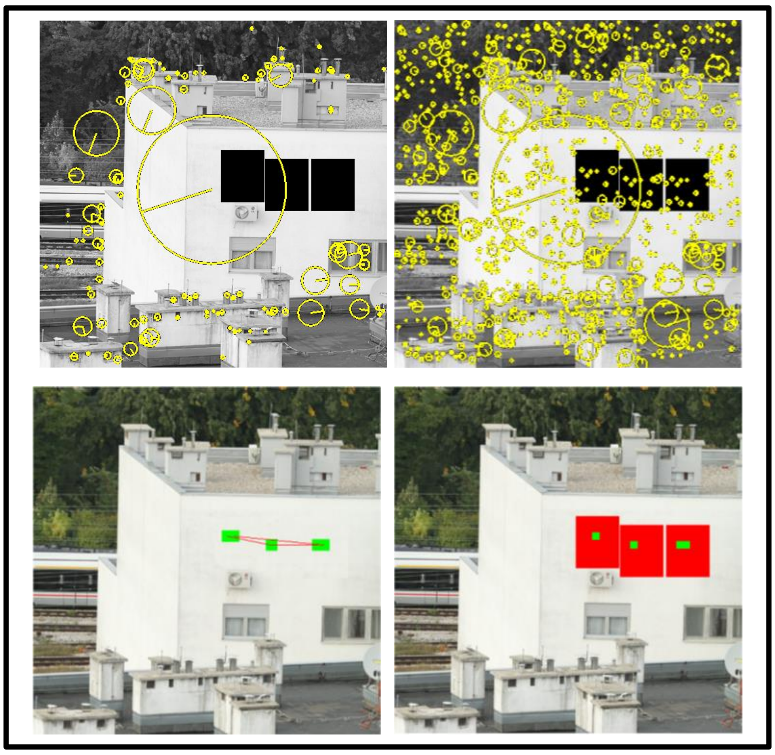

Figure 3-38: detection multiple duplicated objects using Improved SIFT on flat region (top to bottom, left to right) SIFT keypoints with refinement step, SIFT keypoints with refinement step, primary detection, Final result.

In contrast, the improved DSIFT can detect forgery on the same image with an excellent F-measure (0.99), see Figure 3-39.

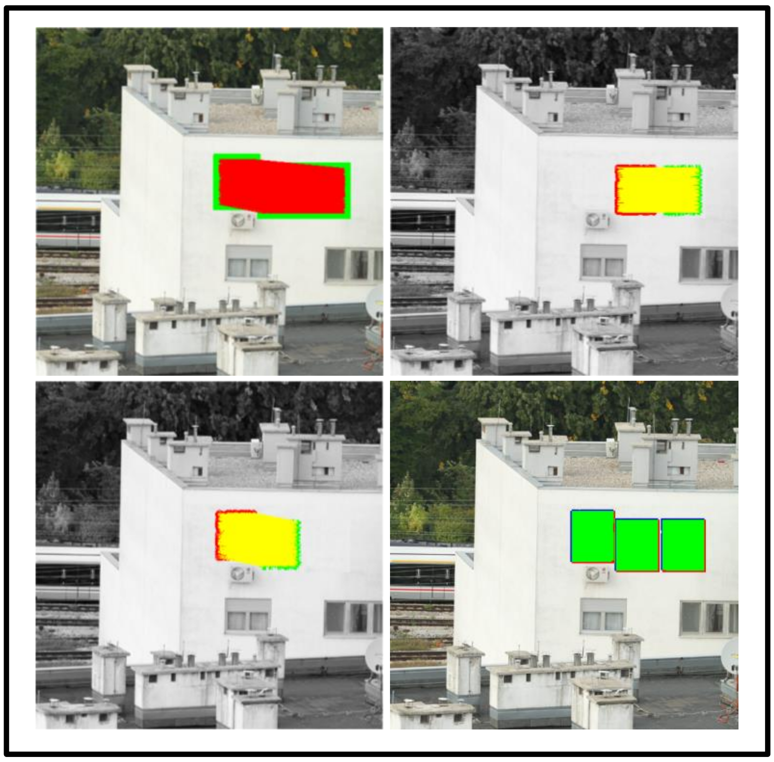

Figure 3-39: detecting multiple duplicated objects using Improved DSIFT on flat region: (top to bottom, left to right) primary detection, RANSAC result, RANSAC result, final result.

2. SIFT's ability to partially detect Copy-Move objects

As noted above, the keypoints of the SIFT are distributed sparsely on the image and do not cover every one of its pixels. Thus matching of these keypoints may not cover all the duplicated objects, see Figure 3-40.

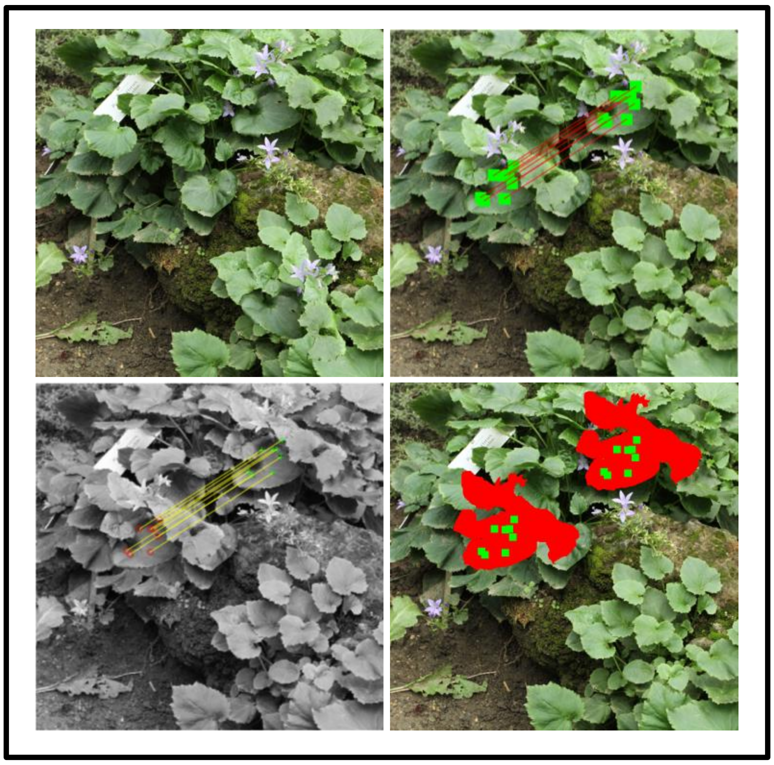

Figure 3-40: detecting multiple duplicated objects using improved SIFT on flat region (top to bottom, left to right), SIFT keypoints with refinement step, SIFT keypoints with refinement step, primary detection, Final result.

3. SIFT estimates two dominant orientations for 15% of keypoints:

As previously claimed in section 3.3.1, if SIFT estimated two dominant orientations for 15% of the keypoints it would mean that the number of generated feature vectors increased by 15%. If the same approach was used in DSIFT, it would considerably increase the computation time spent on the features extraction and matching.

We used the standard moments (intensity centroid measure) [93] to estimate the canonical orientation for each patch in the image and thus obtained results which are more accurate and have better noise resistance ability than those which are gradient based orientation.

3.11 Conclusions

SIFT is the most widely used descriptor; it is discriminative and relatively fast. As previously discussed in Section 3.10, using SIFT for detecting CMF has three main drawbacks (SIFT rejects keypoints in flat regions, SIFT’s ability to only partially detect copy-move objects, SIFT estimates two dominant orientations for 15% of keypoints). To overcome these drawbacks, we have combined ideas derived from the keypoint and block-based methods. We applied SIFT densely to make block-based matching possible and cover all the pixels of the image. Also, we used the intensity centroid measure method, to estimate patch orientation, instead of standard SIFT method, which makes our improved DSIFT rotation invariant without multiple orientation assignment for the same patch. Our proposed improved DSIFT is more robust to rotation than RIDSIFT and Zernike moments, see Sections 3.3.3 and 3.9.4.

SIFT considers a square region (16×16) around the keypoint, which introduces border. We eliminate the border effect by using a circular block of radius 7.5 in the improved DSIFT. In comparison, using a circular shaped Gaussian weighting in the standard SIFT can only reduce the effects of square blocks, see Section 3.3.3.

Moreover, experimentally we found that the improved DSIFT is highly discriminative and can detect forgery in flat regions and complex structured images.

The similarity threshold between feature vectors is one of the most important parameters in detecting CMF. Consider that this threshold depends on the image characteristics (e.g. the texture, the colour distribution, and the edges) and varies from one image to another. Using a single threshold for many images in the dataset either produces a high false positive when choosing a large value, or high false negative when using a small value. Our proposed method to compute the automatic thresholding reduces the false positive and decreases the required time to estimate one threshold for different images in the dataset. Also, the hysteresis technique was used to grow the detection regions and improve the primary detection result which reduces the false negative. The use of the automatic thresholding with the hysteresis technique can produce the best possible result, see Table 3-4.

Chapter 4
Copy-Move Forgery Detection with a Surface Fitting Method

4.1 Introduction

As previously discussed in Section 2.4.1, an exhaustive search is the simplest approach for detecting copy-move forgery in an image, but it is time-consuming and cannot detect complex transformed copy-moved objects [25].

The standard CMF detection approach first extracts features from the image, then compares the features to find the copy-move parts. There are two possible methods of extracting features from an image: either a sparse keypoint-based method or a dense block-based method. Keypoint-based methods consume very little memory and are much faster than block-based methods. However, keypoint-based methods can detect only parts of the copy-moved objects or produce a false negative in flat regions. Conversely, block-based methods require high computational time for extracting and matching the features.

If we consider the image as a surface, then theoretically the surfaces of the copy-moved objects have similar curvatures. In many geometry processing applications, finding matches between two surfaces is a standard task. Still, the existing approaches to this task, such as matching two meshes directly in 3D space, exact a high computational cost [109]. To overcome the problem of high computation, we propose here a novel method for finding matching between the image blocks; it uses the least squares method to fit a quadratic function on each block, and uses the parameters of the fitted function to represent the shape of the surface on each block. Then we find the matches between these feature vectors (parameters).

4.2 Surface Fitting for Copy-Moved Block Detection

An image can be represented by a continuous function of two variables $f(x,y)$, where (x,y) are the coordinates of the image plane [110]. Copy-moved blocks can be represented by a similar function. Theoretically, the surfaces (curvatures) of the copy-moved objects are similar (see Figure 4-1), and fitting a function on the surfaces of these blocks will generate similar parameters.

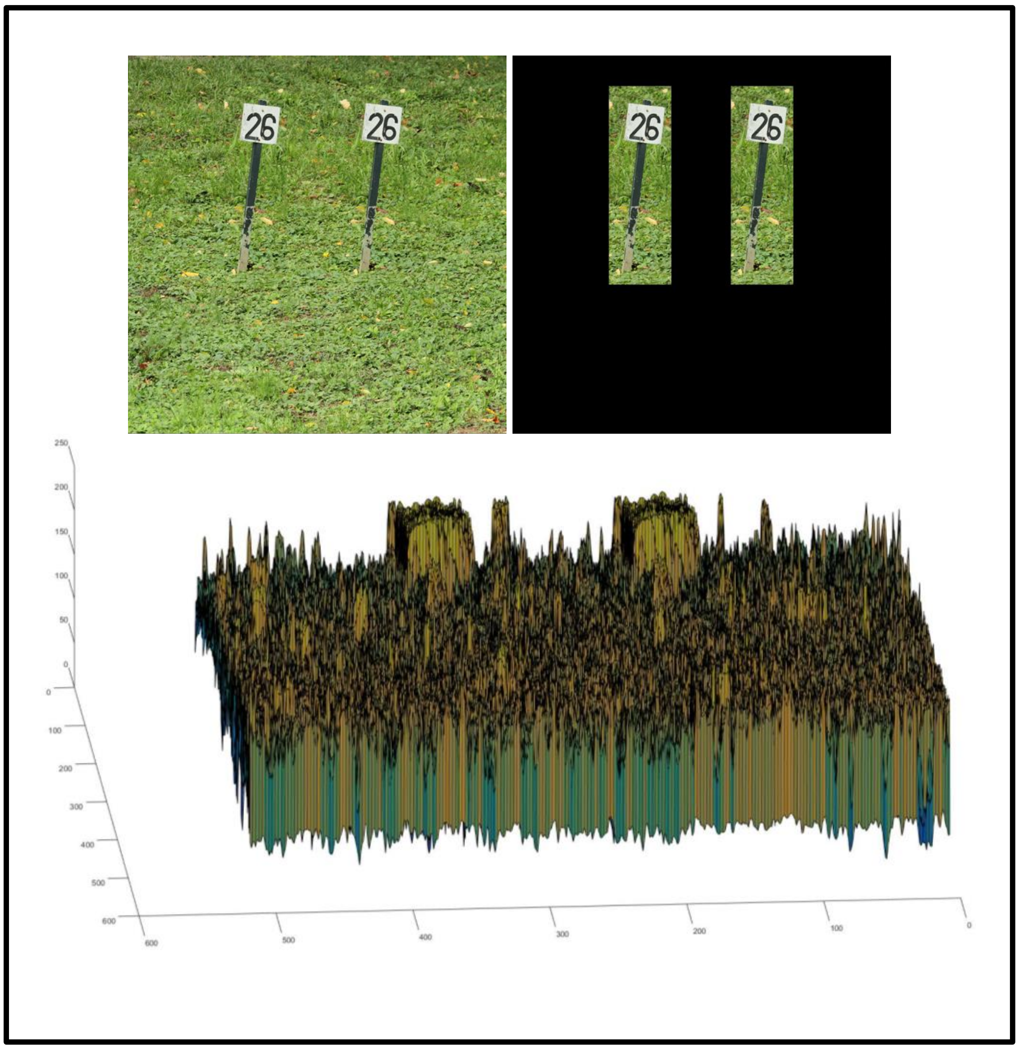

Figure 4-1: (left to right, top to bottom) the input forged image, the Copy-Moved objects and the image as surface.

Let B_1 and B_2 be a copy-move pair of blocks with centres (x_1, y_1) and (x_2, y_2), respectively. The difference between the fitted function of these two blocks is very small and less than a specific threshold;

$$|f_1(B_1) - f_2(B_2)| \leq threshold \qquad \textbf{(4-1)}$$

For simplicity, we use a bivariate quadratic function to represent the surfaces of the blocks, and compare the parameters of the fitted functions to find the copy-moved blocks. These parameters can be estimated by using the least squares method, it determines the straight-line/curve that best fits the observed data points [111].

4.3 Fitting Quadratic Function using the Least Squares Method

To extract the features (parameters) of an image block, we first convert the colour image into grayscale and then divide it into (17×17) overlapping blocks, see Figure 4-2. The least squares method is used to fit the parameters of the bivariate quadratic function representing the surface of each block. Suppose I is a grayscale image and (i, j) is the centre of an image block. The gray level values of the block can be represented by a matrix M.

$$M = \begin{bmatrix} I_{i-8,j-8} & \cdots & I_{i+8,j-8} \\ \vdots & \ddots & \vdots \\ I_{i-8,j+8} & \cdots & I_{i+8,j+8} \end{bmatrix}$$

As previously established, a 2D quadratic function can be used to represent each block. Let (u, v) represent the coordinates in the block's coordinate system with its origin being the block centre; then a 2D quadratic function can be represented by

$$f(u, \mathrm{v}) = \mathrm{a} + bu + cv + du^2 + euv + fv^2 \qquad \textbf{(4-2)}$$

and the matrix M can be formed as follows

$$M = a \cdot \mathbf{1} + b \cdot U + c \cdot V + d \cdot U_2 + e \cdot P + f \cdot V_2 \qquad \textbf{(4-3)}$$

where

$$\mathbf{1} = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}$$

$$U = \begin{bmatrix} -8 & \cdots & 8 \\ \vdots & \ddots & \vdots \\ -8 & \cdots & 8 \end{bmatrix}$$

$$V = \begin{bmatrix} -8 & \cdots & -8 \\ \vdots & \ddots & \vdots \\ 8 & \cdots & 8 \end{bmatrix}$$

$$U_2 = \begin{bmatrix} 64 & 49 & \cdots & 49 & 64 \\ 64 & 49 & & 49 & 64 \\ \vdots & & \ddots & & \vdots \\ 64 & 49 & \cdots & 49 & 64 \\ 64 & 49 & & 49 & 64 \end{bmatrix}$$

$$P = \begin{bmatrix} 64 & 56 & \cdots & -56 & -64 \\ 56 & 49 & & -49 & -56 \\ \vdots & & \ddots & & \vdots \\ -55 & -49 & \cdots & 49 & 56 \\ -64 & -56 & & 56 & 64 \end{bmatrix}$$

$$V_2 = \begin{bmatrix} 64 & 64 & \cdots & 64 & 64 \\ 49 & 49 & & 49 & 49 \\ \vdots & & \ddots & & \vdots \\ 49 & 49 & \cdots & 49 & 49 \\ 64 & 64 & & 64 & 64 \end{bmatrix}$$

Rearranging the seven matrices $(M, 1, U, V, U_2, P, V_2)$ as vectors by concatenating the columns of each matrix, Equation (4-2) can be rewritten as $Y = X\theta$, where $Y \in R^{289}$ is the rearrangement of M, $X \in R^{289\times 6}$ consists of the rearrangement of $(1, U, V, U_2, P, V_2)$, and $\theta \in R^6$ where $\theta = [a, b, c, d, e, f]^T$ is the parameter vector.

The least squares method gives the estimate of parameter vector θ as

$$\hat{\theta} = (X^T X)^{-1} X^T Y \qquad \textbf{(4-4)}$$

We can thus use $\hat{\theta}$ as the block features and match these features to find the copy-moved parts.

To detect copy-rotate-move objects, we need to make the features rotation invariant before feature matching.

The method we use to make the feature rotation invariant is to rotate the block surface so that the principal axes of the quadric surface are parallel to the image coordinate axes; i.e. the parameter e of the cross-axis term $u\ v$ is zero.

Using matrices and vectors to represent Equation (4-2), we have

$$f(u,v) = a + \theta_1^T \cdot \boldsymbol{x} + \boldsymbol{x}^T\ \Theta_2\ \boldsymbol{x} \qquad \textbf{(4-5)}$$

where $\boldsymbol{x} = \begin{bmatrix} u \\ v \end{bmatrix}, \theta_1 = \begin{bmatrix} b \\ c \end{bmatrix},\ \Theta_2 = \begin{bmatrix} d & e/2 \\ e/2 & f \end{bmatrix}$

Suppose a point (u, v) is rotated by angle $\theta,$ then the new position $(\acute{u}, \acute{v})$ can be represented by

$$\begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

So, we have

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix}$$

For simplicity, we will refer to $\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$ as R

We substitute $\begin{bmatrix} u \\ v \end{bmatrix}$ by $\begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix}$, which generates the following:

$$f(\acute{u}, \acute{v}) = a + \theta_1^T\ R\ \begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix} + \begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix}^T R^T \Theta_2\ R \begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix} = a + \acute{\theta}_1^T \begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix} +$$

$$[\acute{u} \quad \acute{v}]\ \acute{\Theta}_2 \begin{bmatrix} \acute{u} \\ \acute{v} \end{bmatrix} \qquad \textbf{(4-6)}$$

where $\acute{\theta}_1 = R^T \theta_1,$ and $\acute{\Theta}_2 = R^T \Theta_2\ R$

To eliminate cross-axis terms in the new representation (4-6), we need $\acute{\Theta}_2$ to be a diagonal matrix.

We can get a diagonal $\acute{\Theta}_2$ by Eigen decomposition of Θ_2

$\Theta_2 = R \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R^T$ where (λ_1 λ_2) are eigenvalues and R is the matrix consisting of the eigenvectors.

$$R^T R = \; R\, R^T = 1$$

Thus, we get $\acute{\Theta}_2 = R^T \Theta_2\, R = \; R^T R \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R^T R = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$.

Now, we can use the parameters $a, \acute{\theta}_1^T = [\acute{b}, \acute{c}], and\ \lambda_1\ \lambda_2$ to form a feature vector $\acute{\Theta}^T = [a, \acute{b}, \acute{c}, \lambda_1\ \lambda_2]$.

Because we do not want brightness changes to affect the detection of CMF, we remove parameter a from the parameter vector Θ́ and the remaining 4-dimensional parameter vector $[\acute{b}, \acute{c}, \lambda_1\ \lambda_2]$ is taken as the feature vector. We call this feature the Fitting Function Rotation Invariant Descriptor (FFRID) and use it for CMF detection.

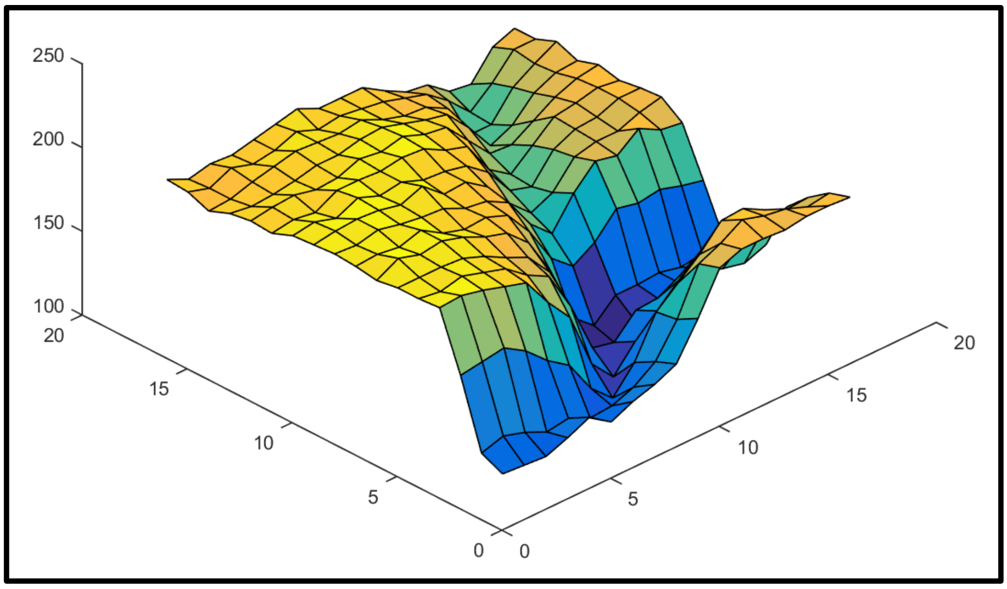

Figure 4-2: an example of a (17x17) block surface.

4.4 High-level description of the proposed algorithm to detect CMF using Fitting Function

This is a full description of the suggested algorithm for detecting CMF using the Fitting Function which is essentially the same as the improved DSIFT version (section 3.5.1), but replacing improved DSIFT with FFRID.

1) Convert the image to grayscale if it is a colour image.
2) FFRID, instead of improved DSIFT, is used to compute the OptimumThreshold using our proposed method as described in Section 3.4.
3) Divide the image into overlapping blocks (17×17)

   If the Median Absolute Deviation (MAD) of the tested block is bigger than Threshold_0=0.09.

      i. Compute the Fitting Function Rotation Invariant Descriptor for the tested block.
      ii. Save the feature vector and the coordinates of the tested block.

   End if
4) Build a k-d tree for the feature vectors and find the 2nd Approximate Nearest Neighbours for each element in the tree.
5) Find the matched feature vectors (blocks) which satisfy:
   a. The Euclidean distance between feature vectors is less than $\text{Threshold}_1 = (\text{OptimumThreshold}/2)$.
   b. The distance between the centres of the two matched blocks being bigger than $\text{Threshold}_2 = 17\sqrt{2}$.
6) Save the coordinators of the two matched blocks in a List.
7) Use the neighbourhood clustering method, proposed in Section 3.7 to remove the false matching from List.

8) Call RANSAC to estimate the transformation of the List.
9) Use hysteresis technique, proposed in Section 3.8, to grow regions of the List.
10) Remove small areas with less than (Threshold_3=320 pixels) in size.
11) Mark the blocks as matching.
12) Morphologically close the image.

4.5 The Experiments

We used CoMoFoD dataset (small images) [23] to evaluate our algorithms.

4.5.1 CMF detection with Translation and postprocessing

We tested 40 different images with plain CMF and successfully detected forgery in all of them. Then we used our suggested method to test 240 images with different types of post-processing (image blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise), see Table 4-1.

Table 4-1 CMF detection with different types of post-processing (Avg. of F-Measure).

Post-processing	F measure	Detected images using FFRID
Translation (without post-processing)	0.93	40
Image Blurring,(5x5 average filter)	0.84	39
Brightness Change Range (0.01, 0.8)	0.92	40
Colour Reduction (32 intensity levels)	0.93	40
Contrast Adjustment Range (0.01,0.8)	0.93	40
White Gaussian Noise(AWGN) =0.001	0.82	38
JPEG (40%)	0.90	40

Figure 4-3 shows an example of detecting CMF on complex texture image, Figure 4-4 shows the CMF detection on flat regions and Figure 4-5 on homogeneous regions. Figure 4-6 and 4-7 show the detection of CMF on post-processed images.

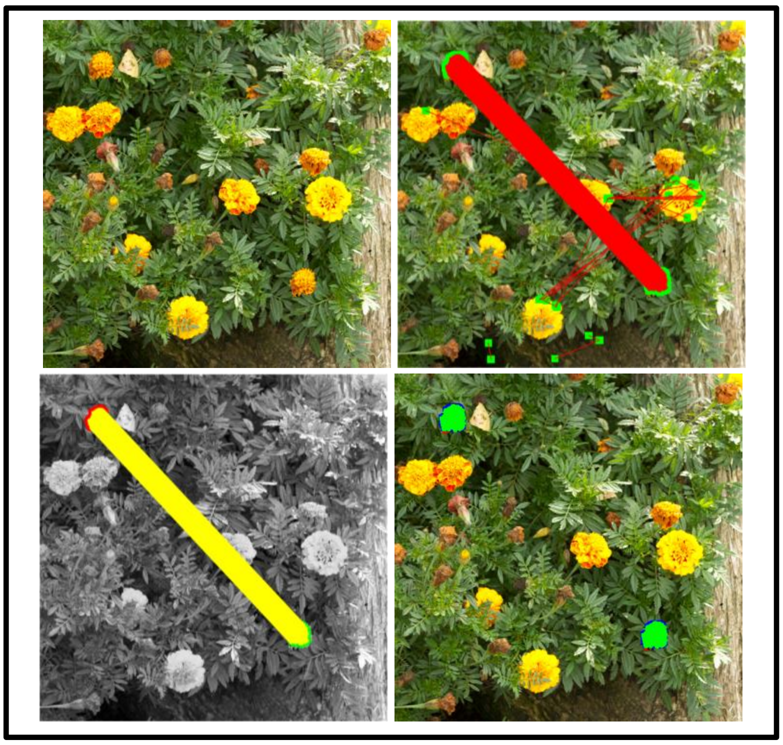

Figure 4-3: An example of using FFRID: (top to bottom, left to right) input forged image, primary detection, RANSAC result, final result after hysteresis.

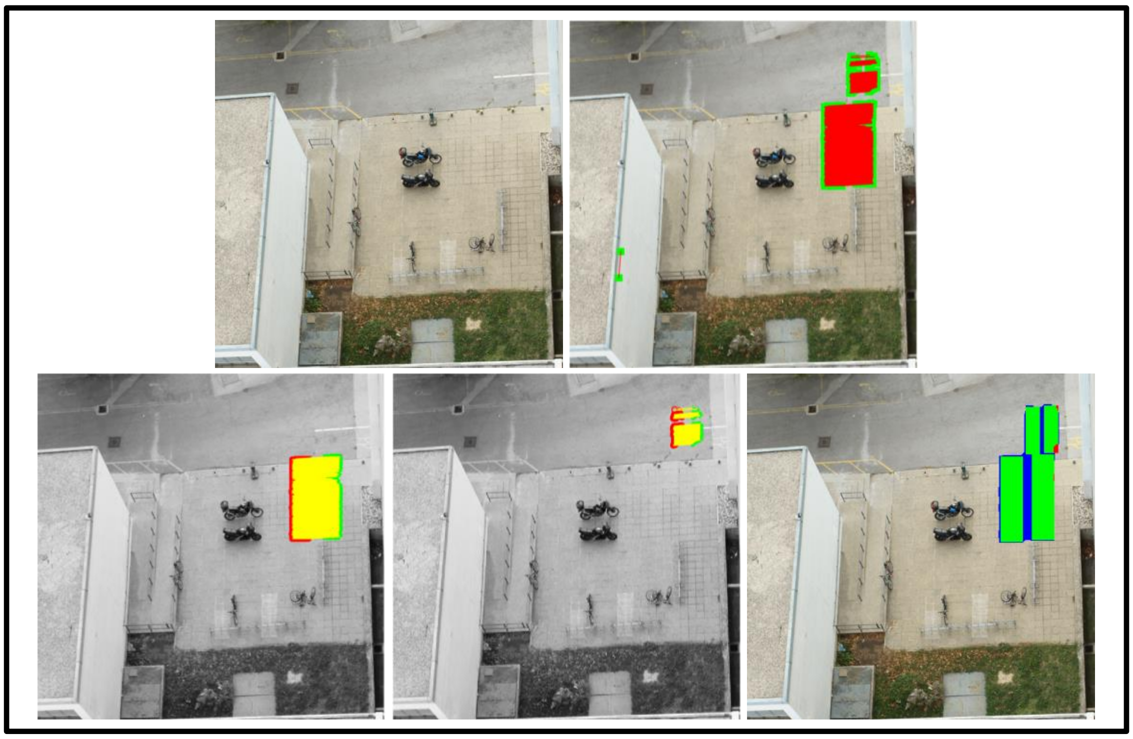

Figure 4-4: An example of detecting multiple duplicated objects using FFRID on a flat image: (top to bottom, left to right) input forged image, primary detection, RANSAC, RANSAC result, final result after hysteresis.

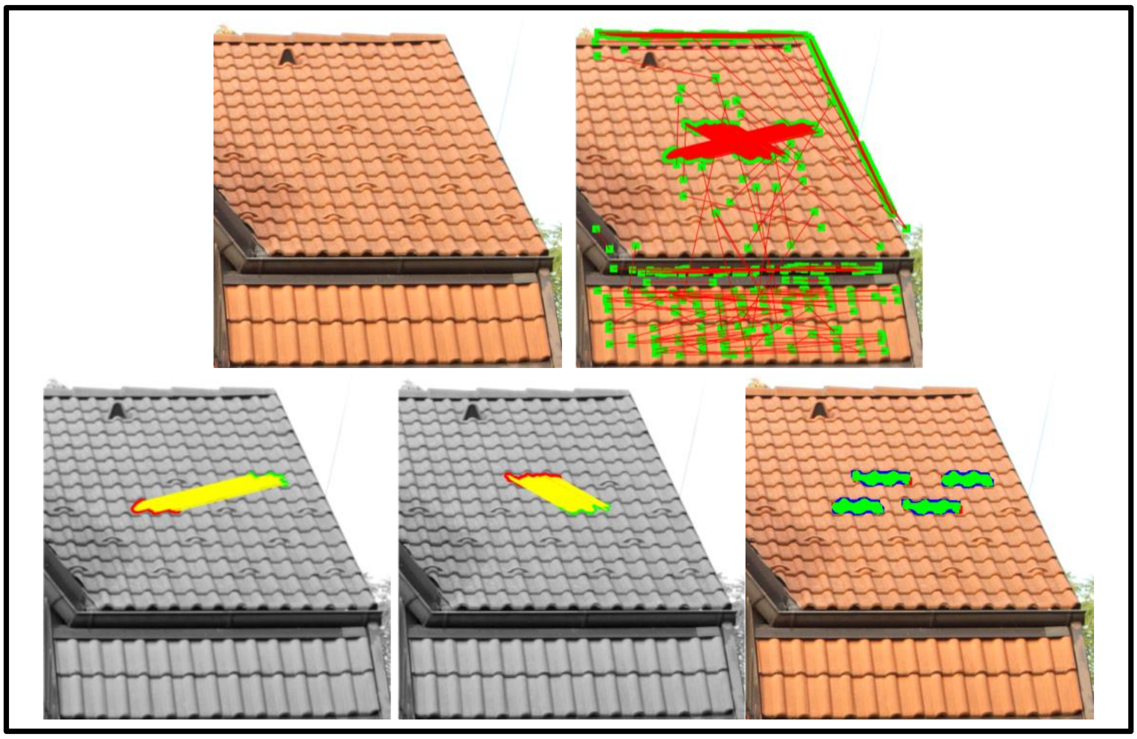
Figure 4-5: An example of detecting multiple duplicated objects using FFRID on a homogeneous image: (top to bottom, left to right) input forged image, primary detection, RANSAC result, RANSAC result, final result after hysteresis.

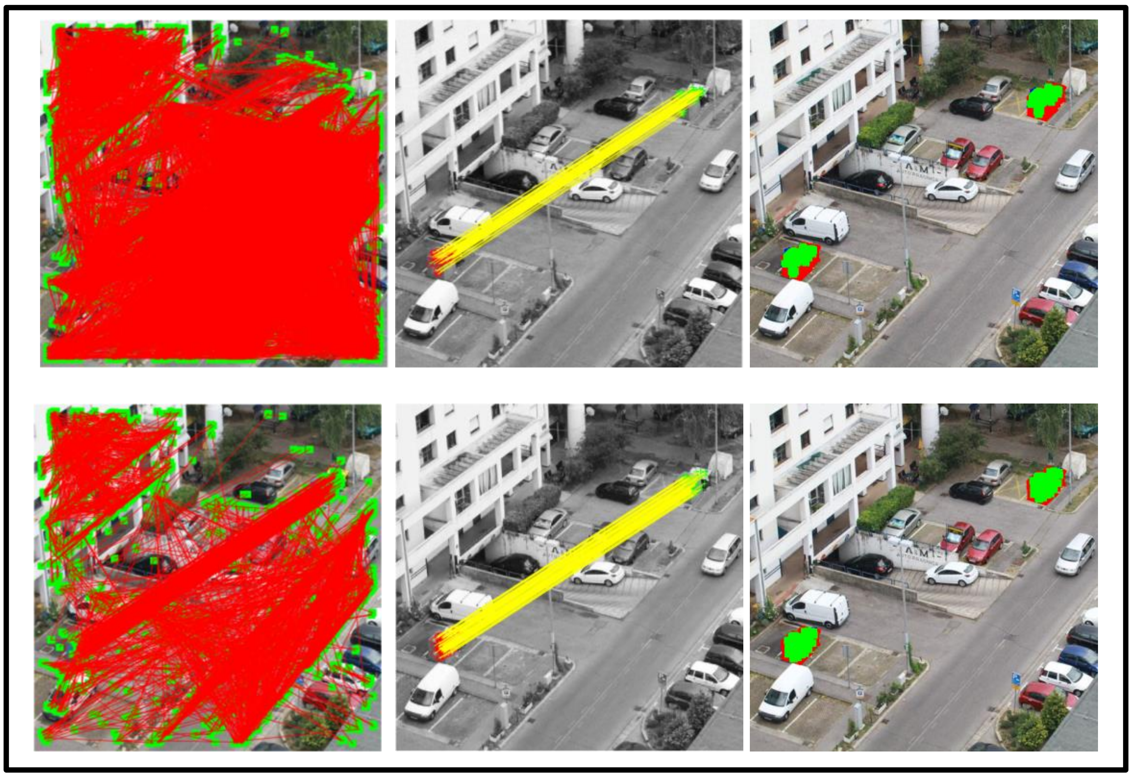
Figure 4-6: An example of using FFRID to detect forgery in post processed images: (1st row) the steps of detection in noisy image, (2nd row) the steps of detection in image with JPEG compressed image.

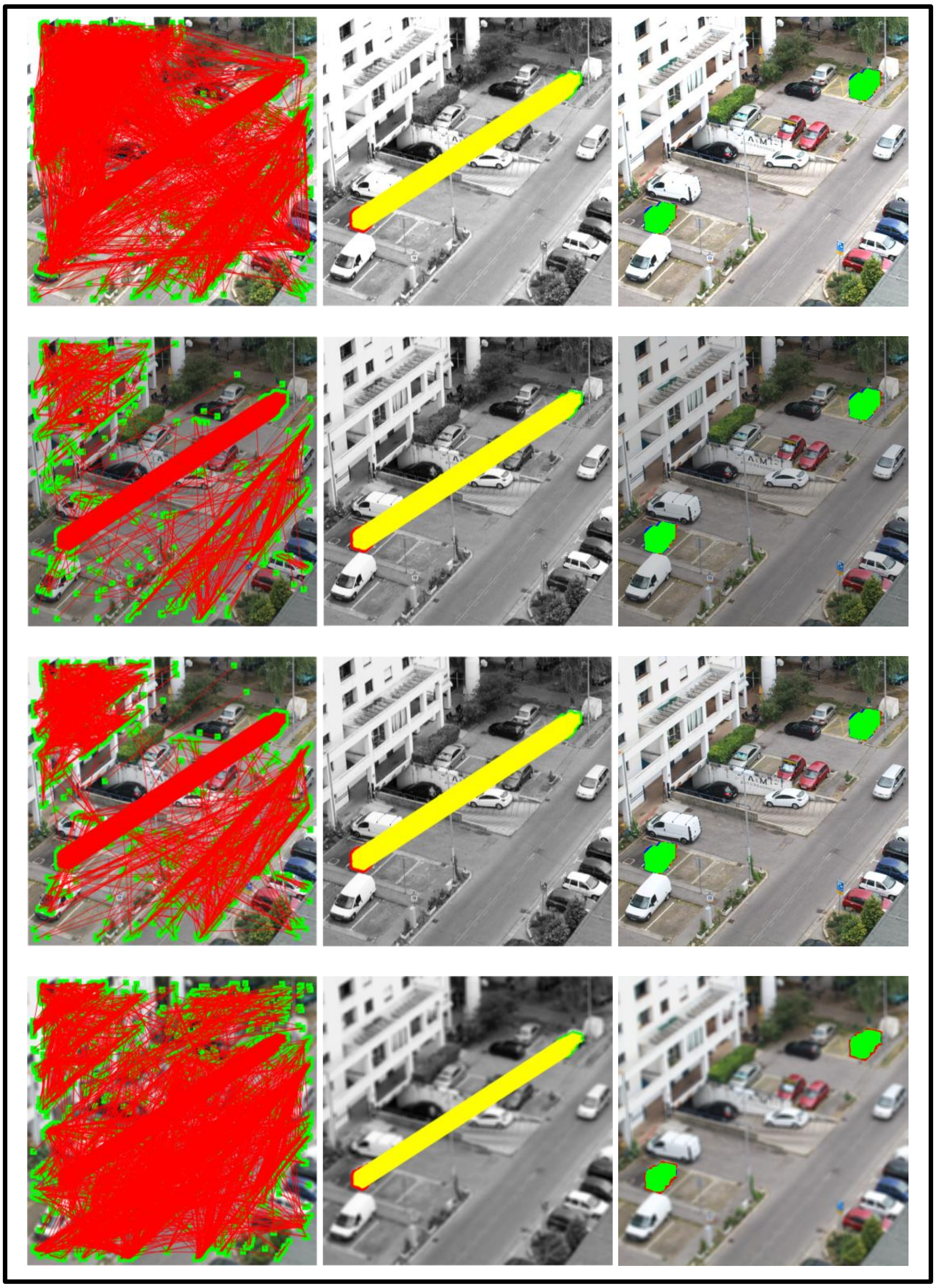

Figure 4-7: An example of using FFRID to detect forgery in post processed images: (1st row) the steps of detection in image with brightness change, (2nd row) the steps of detection in image with Contrast Adjustment, (3rd row) the steps of detection in image with Colour Reduction, (4th row) the steps of detection in image with Image Blurring.

4.5.2 Copy-Rotate-Move Forgery (CRMF) detection

An experiment was conducted on 40 different images with CRMF. We were able to detect forgery on all the tested images where the F-measure was 0.76, see Table 4-2. Below, some examples with different image properties are shown. Figure 4-8 shows an image with homogeneous textures in which it is very hard to detect forgery with the naked eye. In Figure 4-9, the size of the CMF region is very small and it is hard to detect duplicated regions. Figure 4-10 and Figure 4-11 show the detection of CRMF with post-processed images.

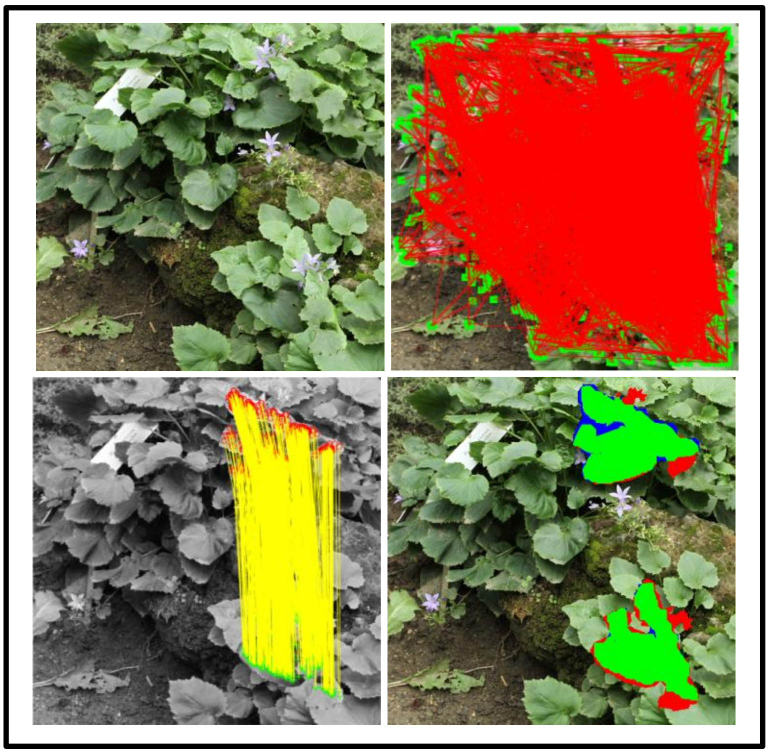

Figure 4-8: An example of detecting duplicated objects using FFRID: (top to bottom, left to right) input forged image, copied object rotated by 40°, primary detection, RANSAC result, final result after hysteresis.

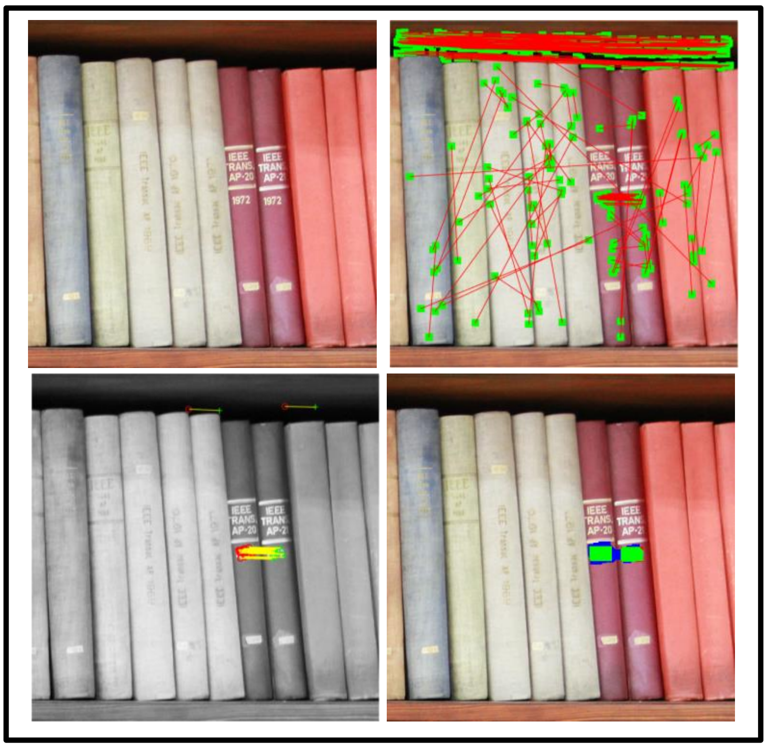

Figure 4-9: An example of detecting duplicated objects using FFRID: (top to bottom, left to right) input forged image, copied object rotated by -1°, primary detection, RANSAC result, final result after hysteresis.

Table 4-2 CRMF detection with different types of post-processing (Avg. of F-Measure).

Post-processing	**F measure**	**Detected images using FFRID**
Rotation (without post-processing)	0.76	40
Image Blurring,(5x5 average filter)	0.67	36
Brightness Change Range (0.01, 0.8)	0.66	36
Colour Reduction (32 intensity levels)	0.65	35
Contrast Adjustment Range (0.01,0.8)	0.63	35
White Gaussian Noise(AWGN) =0.001	0.61	28
JPEG (40%)	0.72	40

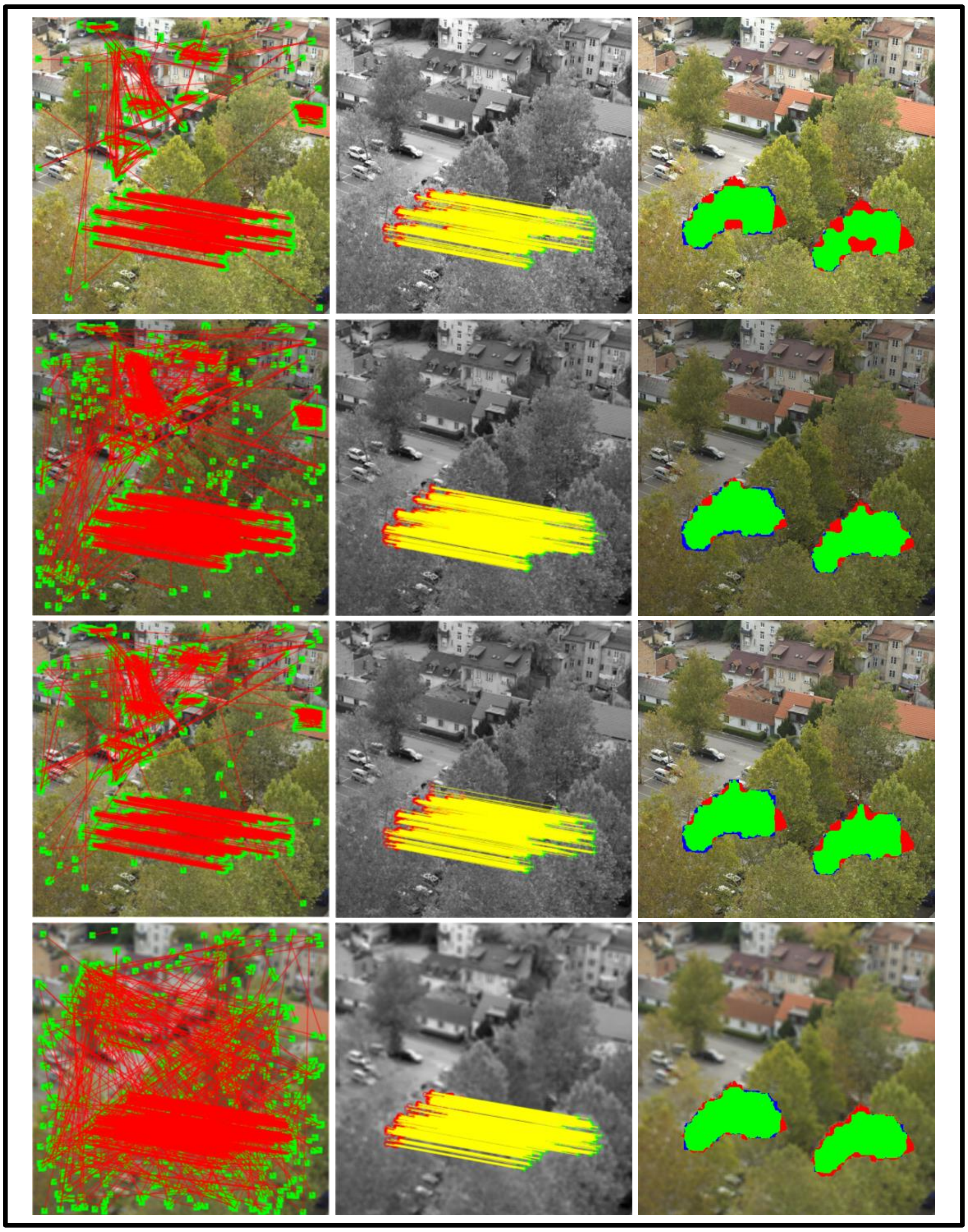

Figure 4-10: An example of using FFRID to detect forgery in post processed images: the copy move objects rotated by 2°, (1st row) the steps of detection in image with brightness change, (2nd row) the steps of detection in image with Contrast Adjustment, (3rd row) the steps of detection in image with Colour Reduction, (4th row) the steps of detection in image with Image Blurring.

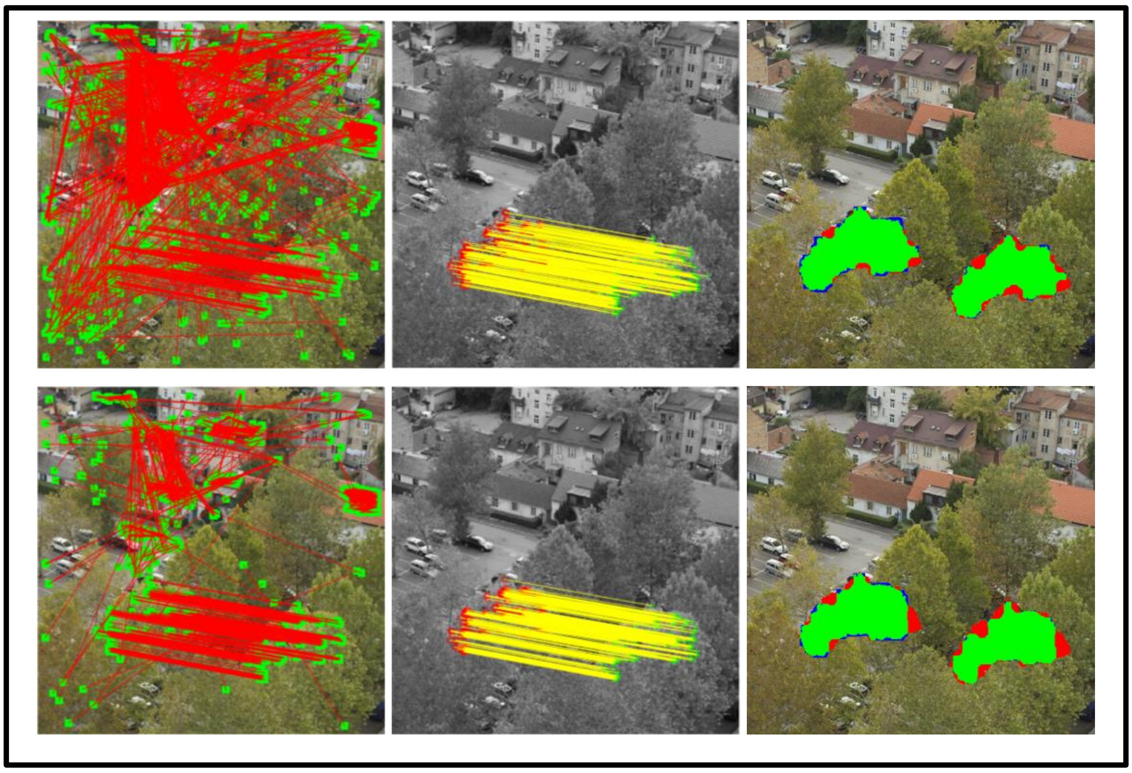

Figure 4-11: An example of using FFRID to detect forgery in post processed images: the copy move objects rotated by 2°, (1st row) the steps of detection in noisy image, (2nd row) the steps of detection in image with JPEG compressed image.

4.5.3 Copy-Move Forgery with Scaling

The FFRID is rotation invariant, and is not designed to be scale invariant. We conducted an experiment to test the robustness of FFRID to scale. The overall F-measure is 0.35 for 40 different images with different scale factors (40%-150%). The FFRID with the proposed algorithm detected forgery on 24 images out of 40 with various scaled duplicated regions, see Figure 4-12 where the radius of each disk represents the square root of the number of images.

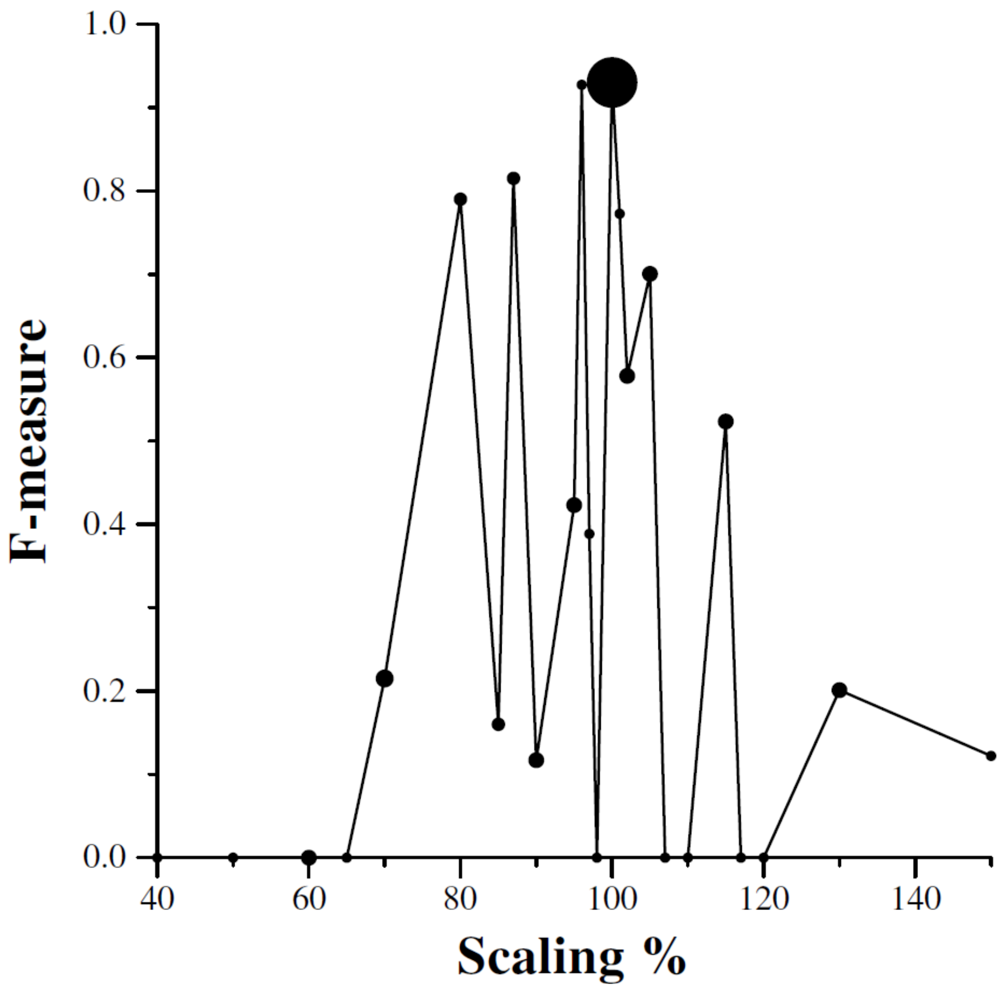

Figure 4-12: Average F-measure for 80 images with different scaling factors.

Experimentally, we illustrate that the FFRID is robust to small changes in scale. The descriptor/the algorithm succeeded in detecting forgery on many different images with moderate scale factors. Below are some examples of detecting duplicated regions with different scales, see Figure 4-13, 4-14 and 4-15. Figure 4-16 shows an example of a failure of detection CMF. Consider that our proposed method can detect forgery on the tested image (primary detection) with high false matches, but it cannot remove the outliers.

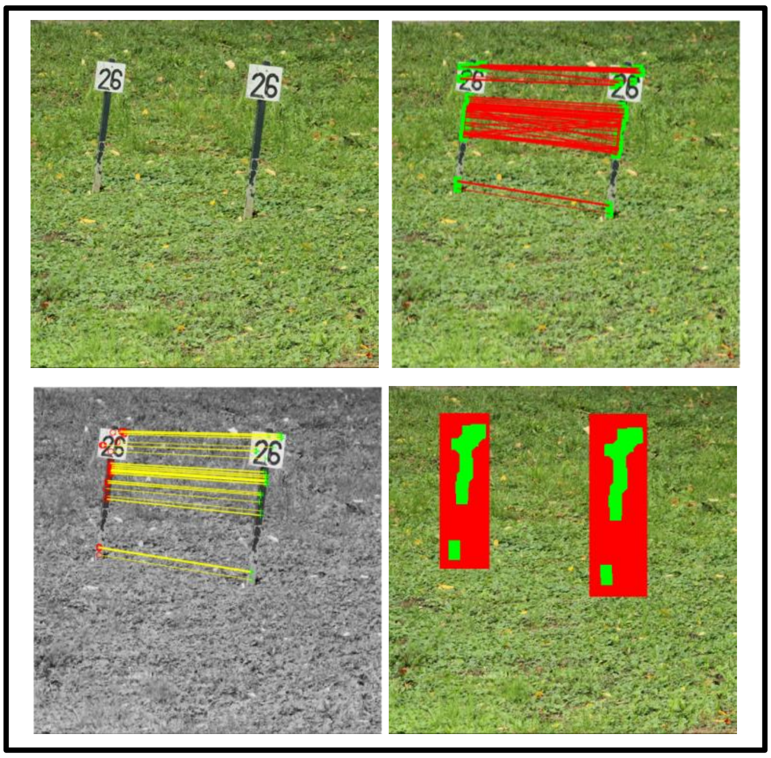

Figure 4-13: An example of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object scaled down by 15%, primary detection, RANSAC result, final result after hysteresis.

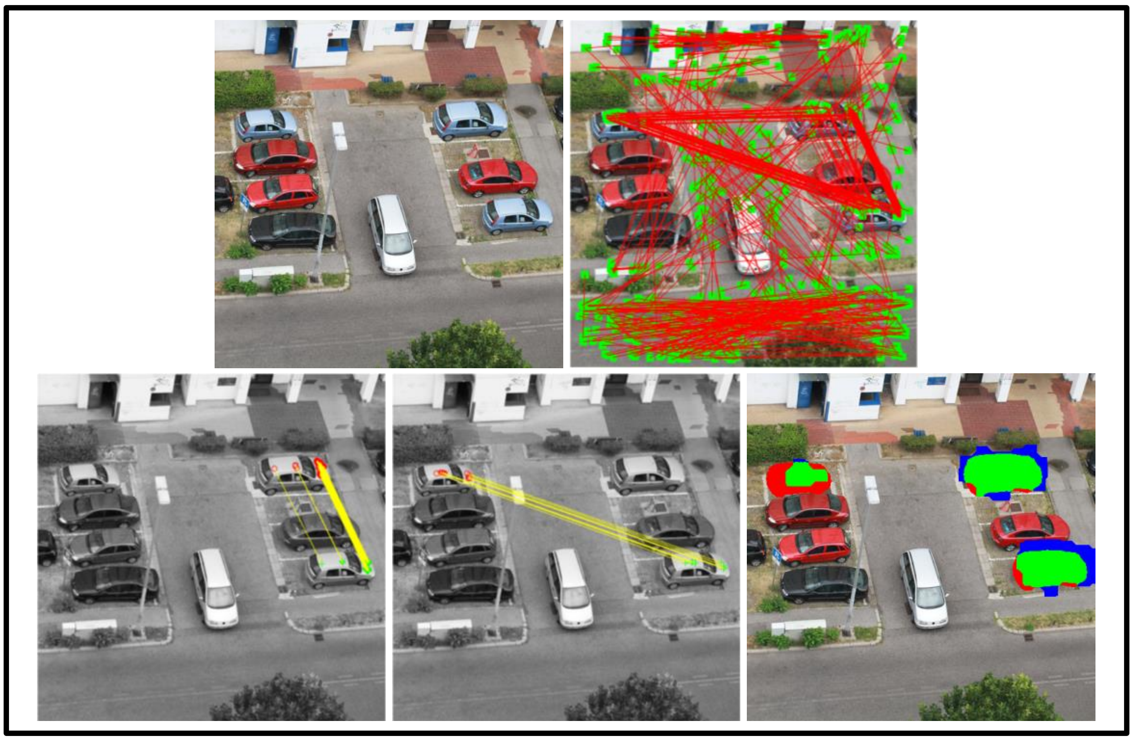

Figure 4-14: An example of detecting multiple duplicated objects by FFRID: (top to bottom, left to right) input forged image, copied object scaled down by 13% and scaled up by 5%, primary detection, RANSAC result, RANSAC result, final result after hysteresis.

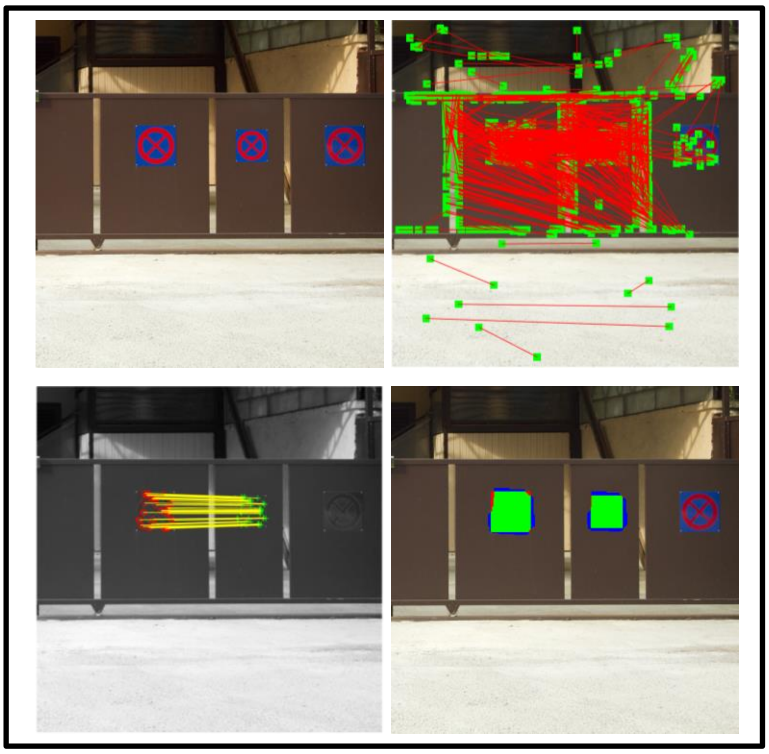

Figure 4-15: An example of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object scaled down by 20%, primary detection, RANSAC result, final result after hysteresis.

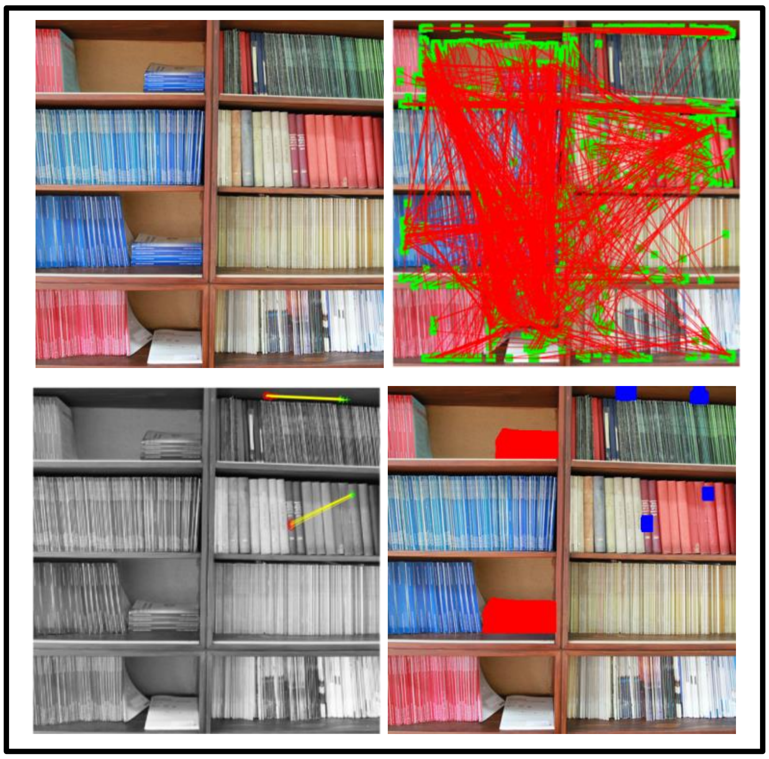

Figure 4-16 An example of failure of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object scaled down by 15%, primary detection, RANSAC result, final result after hysteresis.

4.5.4 Copy-Move Forgery with Distortion

We conducted an experiment to test the robustness of FFRID to distortion, and we achieved good results. The overall F-measure was 0.70 for 40 different images with different distortion factors, see Table 4-3.

Moreover, the FFRID with the proposed algorithm detected forgery on 38 images out of 40 with various distorted duplicated regions, see Figure 4-17 Figure 4-18 and Figure 4-19. Consider that our proposed method can detect forgery on some images (primary detection) but it cannot remove the false matches, e.g. see Figure 4-20.

Table 4-3 CMF detection with different types of distortion (Avg. of F-Measure).

Distortion type	F-measure	Number of detected images out of 10
Vertical scale (80%- 125%)	0.62	10
Horizontal scale (80%-115%)	0.65	9
Vertical skew (1%-8%)	0.73	9
Horizontal skew (1%-7%)	0.82	10

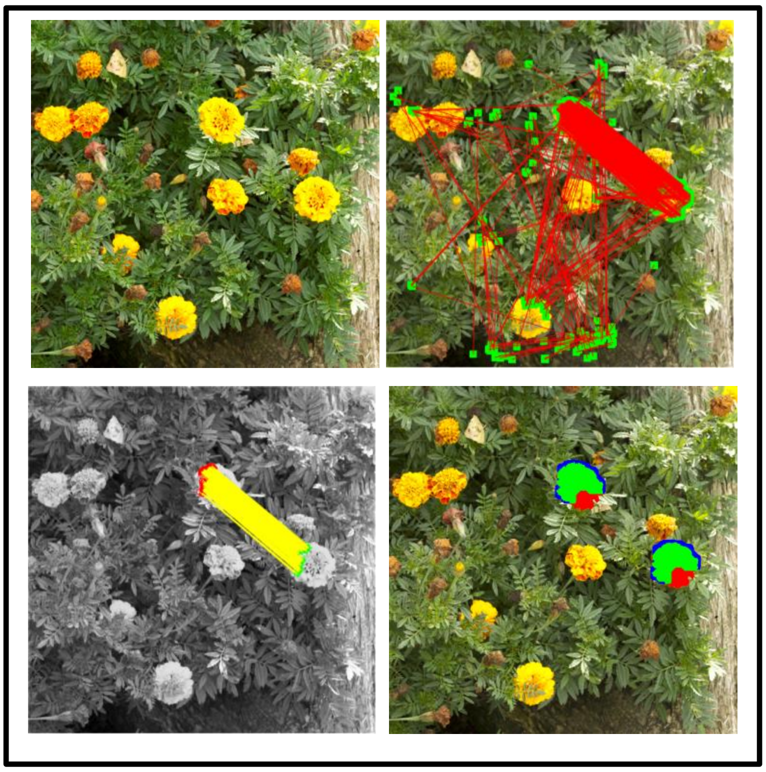
Figure 4-17: An example of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object with a vertical skew factor of 8%, primary detection, RANSAC result, final result after hysteresis.

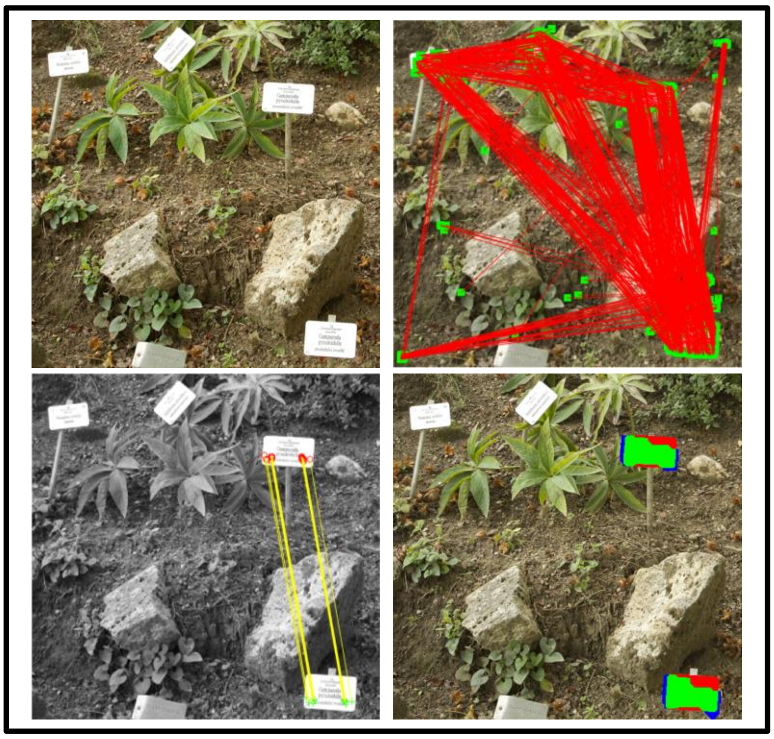

Figure 4-18: An example of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object is vertically elongated by 15%, primary detection, RANSAC result, final result after hysteresis.

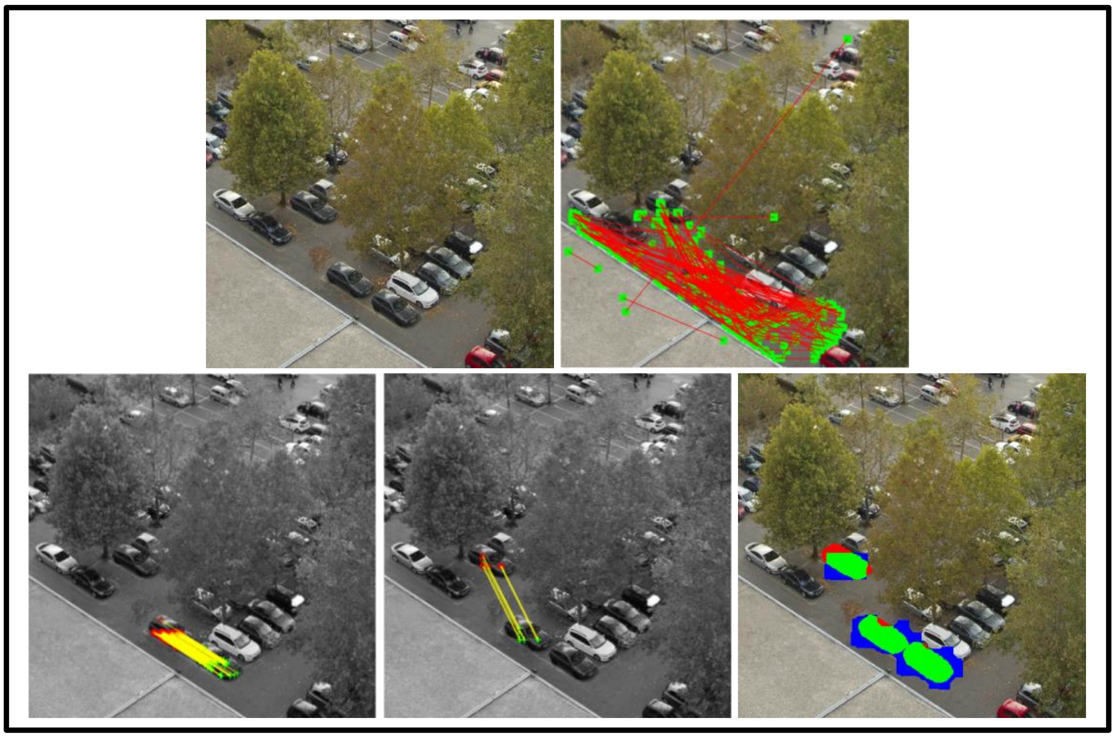

Figure 4-19: An example of detecting multiple duplicated objects using FFRID: (top to bottom, left to right) input forged image, copied object vertically elongated by 5% and by -10 %, primary detection, RANSAC result, RANSAC result, final result after hysteresis.

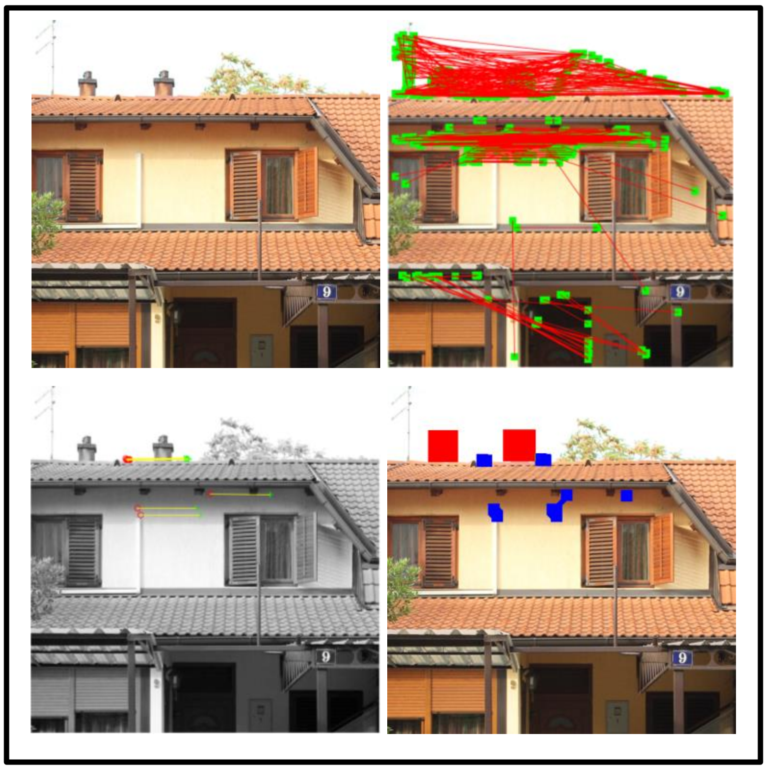
Figure 4-20 An example of failure of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object is horizontally elongated by 10%, primary detection, RANSAC result, final result after hysteresis.

4.5.5 Copy-Move Forgery with Combined Transformation

An experiment was conducted to test the robustness of the proposed algorithm to combined transformations, and we achieved acceptable results. The overall F-measure was 0.45 for 40 different images with different combination factors. Moreover, the proposed algorithm detected forgery on 26 images out of 40 with various combined transformations of duplicated regions, see Figure 4-21, Figure 4-22, Figure 4-23 and Figure 4-24.

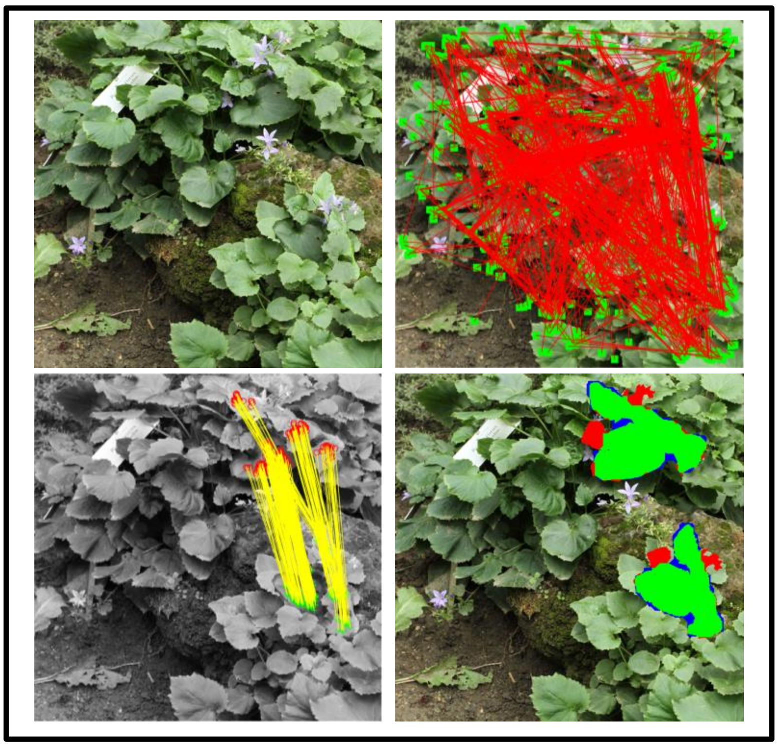

Figure 4-21: An example of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object scaled down by 15% and rotated by 45°, primary detection, RANSAC result, final result after hysteresis.

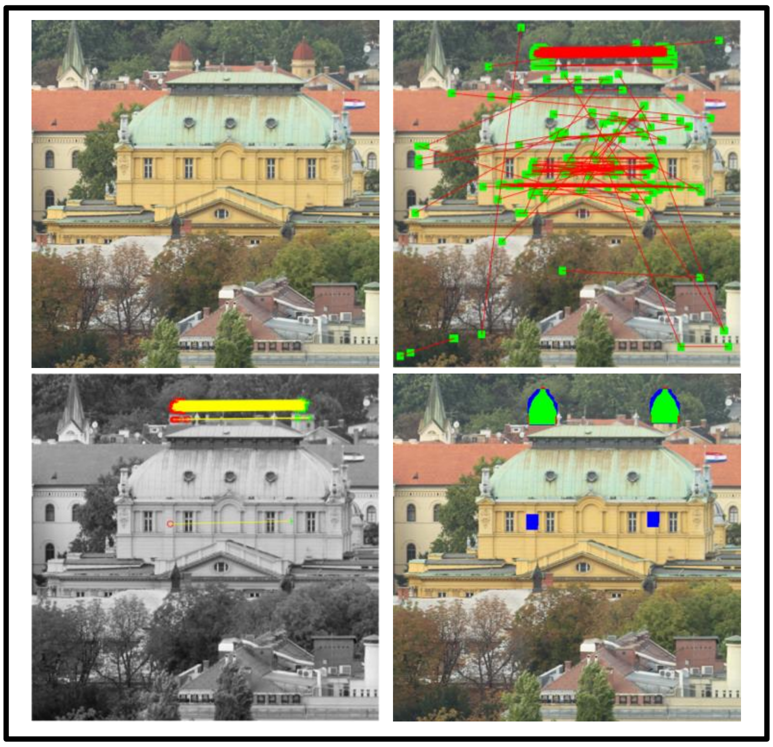

Figure 4-22: An example of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object scaled up by 5% and vertically skewed by 1%, primary detection, RANSAC result, final result after hysteresis.

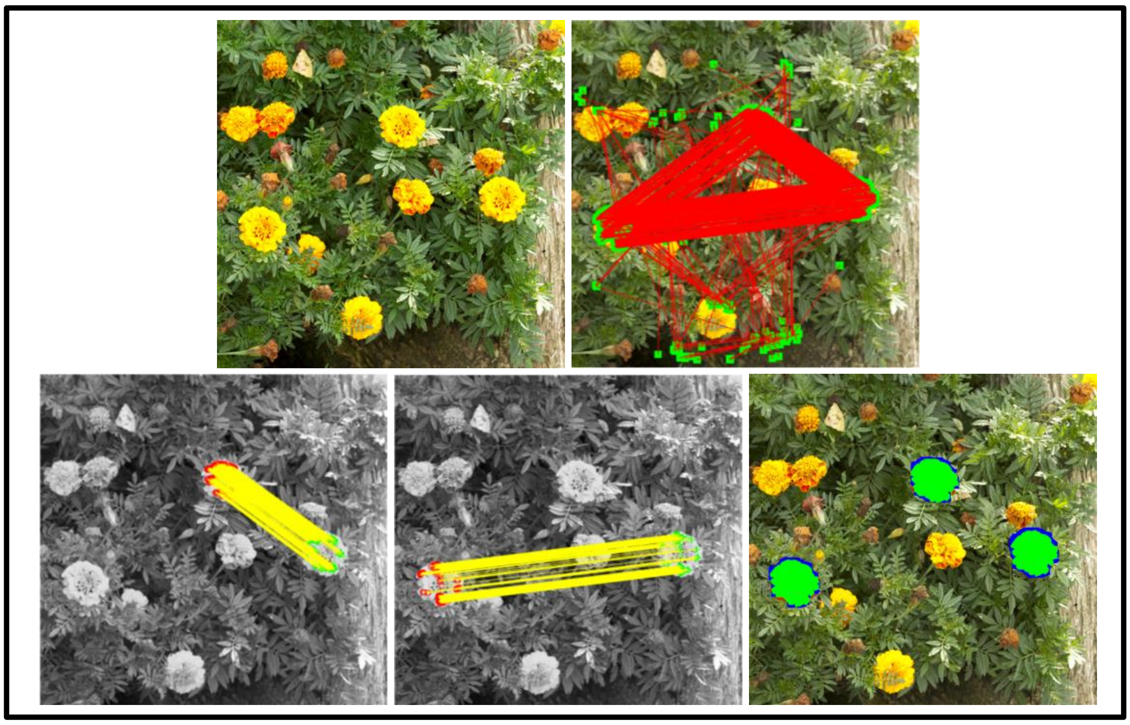

Figure 4-23: An example of detecting multiple duplicated objects using FFRID: (top to bottom, left to right) input forged image, copied objects (horizontally elongated by 3% and vertically skewed by 2%, vertically elongated by -5% and horizontally skewed by 3%), primary detection, RANSAC result, RANSAC result, final result after hysteresis.

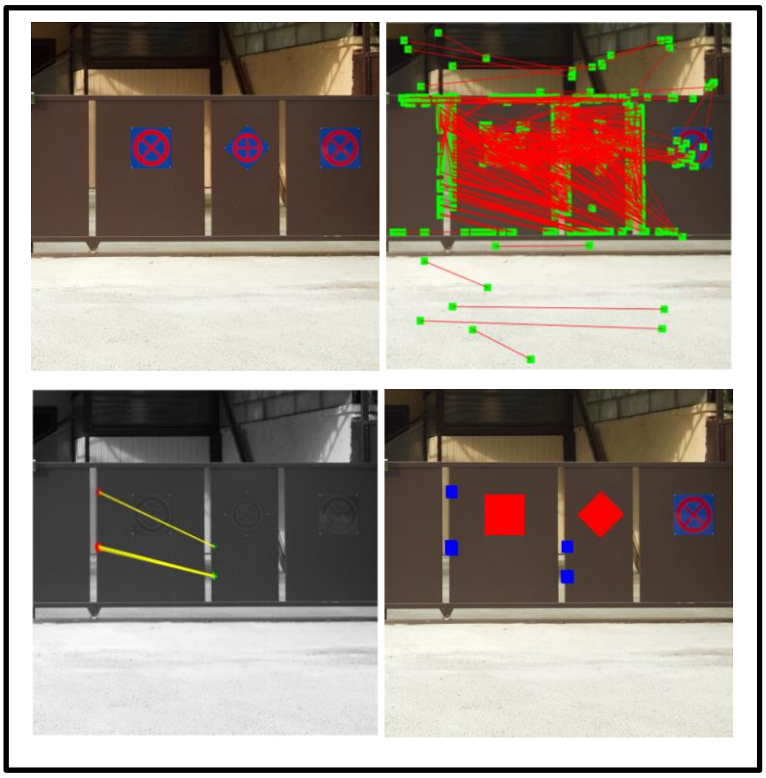

Figure 4-24 An example of failure of detecting duplicated scaled objects using FFRID: (top to bottom, left to right) input forged image, copied object scaled down by 20% and rotated by 45°,, primary detection, RANSAC result, final result after hysteresis.

4.6 Conclusions

As previously mentioned, extracting the features from the image and finding the matches between these features are the most important steps in detecting copy-move forgery. The keypoint method takes a short time to find the interest points in the image, but it cannot cover all the pixels of the image; consequently matching these keypoints may not cover all the duplicated objects.

Conversely, the block-based method covers all the pixels in the image and thus may detect forgery more accurately than the keypoint method. But it has the main drawback of computational cost, which increases significantly with block-based CMF detection. Increasing the number of keypoints/the feature vectors, when they are extracted densely in the block-based method, increases the time required to find the similarities (matches) between these keypoints/feature vectors.

Moreover, the longer the feature vector, the more processing time required to find the similarities; e.g. the high dimensionality of the SIFT descriptor (128 elements) increases the time needed to compute the Euclidean distances between a set of SIFT descriptors.

To overcome these problems we proposed a novel method to describe the shape of the surface of each block in the image. The least squares method was used to fit the six coefficients of the quadratic function on the surface of each block in the image.

CRMF detection requires a rotation-invariant descriptor; thus, we have made the descriptor rotation-invariant. One of the possible methods is rotating the six parameters of the fitted function to be parallel to the principal axes. Therefore, the Eigenvalues and Eigenvectors were computed for the parameters (dx^2, exy and fy^2) which represent the shape of the surface in each block. The parameters (bx, cy) were multiplied by the eigenvectors, to project these parameters around the principal axes, and concatenate them with the eigenvalues. The generated descriptor, which has 4-element feature vector, is rotation-invariant.

The Fitting Function Rotation Invariant Descriptor (FFRID) is faster than the improved DSIFT or Zernike moments; computing the FFRID for an image with of 512×512 pixels takes about 65 seconds. The major loops in the code were vectorised using the array operation in MATLAB, which reduced the time required to compute the descriptors [112]. Moreover, because the FFRID is a 4-element feature vector, finding the potential copy-move objects in the matching step can be done faster than using longer feature descriptors (e.g. DSIFT, HOG).

The proposed descriptor produces excellent results in plain copy-move forgery detection (translation). Different postprocessing methods were used to test the robustness of the descriptor, and in most cases the descriptor maintained its excellent performance. 240 images with plain CMF and with different postprocessing were tested and the proposed method detected forgery on 237 of them.

The FFRID can detect copy-rotate-move forgery with good results, having been designed to be rotation-invariant. Moreover, the FFRID successfully detected forgery on 210 image out of 240 with different postprocessing methods. The FFRID showed an excellent robustness to JPEG compression even in the detection of CRMF objects.

Moreover, our proposed method can detect forgery on many images with complex transformations with high false matches, but it cannot remove those outliers (false matches) on some cases, see Figure 4-16, Figure 4-20 and Figure 4-24.

Chapter 5
Image Segmentation Based Copy-Move Forgery Detection

5.1 Introduction

The most common approach to detecting copy-move forgery takes many steps, the most important being feature extraction. There are two different methods of extracting features, either by tiling the image into blocks and extracting the feature from each block, or extracting it from interest points which are distributed in different ways over the image (e.g. SIFT, MSER, FAST, etc.). The block-based methods usually take a long time to extract the image. The keypoint-based methods are much faster, but they can only detect some part(s) of the duplicated region(s) because it is distributed sparsely over the image.

We consider the segmentation approach as a potential solution to overcome the problems of block-based and keypoint-based methods. We have suggested a new approach to detecting copy-move forgery, in which the detection depends on image segmentation and feature extraction from each segment. The main challenge with this proposed approach is: ***"There is no reliable method of segmenting identical objects consistently."***

Even with a state-of-art segmentation method, there is no guarantee that all the copy-move objects will be segmented in the same manner. This suggests that we should favour over-segmentation as a more uniform method in copy-move forgery detection. Over-segmentation divides the image into non-overlapping and irregular blocks (segments) and we have proposed using the Superpixel method to segment the image and extract some features from each segment.

In this chapter, we tested an existing over-segmentation (SLIC superpixel) based CMF detection method and analysed its performance, see Section 5.2. We further developed two over-segmentation CMF detection methods and analysed their performance, see Sections 5.3 and 5.4.

5.2 Using SLIC superpixel segmentation to detect CMF

Superpixel algorithms cluster pixels into perceptually meaningful regions. SLIC superpixel is the best superpixel algorithm with boundary adherence, computation speed, and performance when used as a pre-processing step in a segmentation framework. Therefore, we used SLIC to segment the images.

The main challenging with this method is the copy-move objects are different in size from one image to another. To overcome this challenging, we used SLIC with 6 different required number of approximately similar-sized superpixels (K) to segment the colour image.

First, we converted the coloured image into grayscale and computed the Local Binary Pattern (LBP) for each pixel to produce the LBP image. Then we built a 256-elements histogram for each 16×16 patch in the LBP image. The K-means++ was used to find 64 centres in the list of LBP feature vector. Then for each different segment in each different scale, we build a 3D RGB colour histogram to describe the colour distribution for each segment. The segment coordinates were used to find the LBP feature vectors which belong to each segment and the Euclidean distance was computed between these feature vectors and each centre. We found the minimum distance between the LBP feature vectors and each centre and built a 64-element Bag of Words Feature Vector (BoWFV). In other words, we used the Bag of Words to quantize the 256-elements histogram into 64 feature vector. We built a K-d tree for the BoWFVs and found the 2nd Approximate Nearest Neighbours within a specific threshold. We built a binary mask for the result of each scale. Then we summed the six binary masks and considered any pixel has value >= 2 as a potential CMF.

5.2.1 High-level description of the proposed algorithm to detect CMF using SLIC superpixel segmentation

1) Convert the image to grayscale if it is coloured.
2) Compute the Local Binary Pattern (LBP) for each pixel in the grayscale image.
3) Build a 256-element LBP Feature Vector (LBPFV) from the histogram of each 16×16 patch in the LBP image.
4) Build the codebook by using the K-means++ to find 64 centres (K=64) in the set of LBP Feature Vectors.
5) Use the SLIC superpixel to segment the coloured image into a different number of segmentations (different scales).

   Number of segmentations = [30, 55, 100, 300, 550, 800], this will generate 6 different segmented images with different scales.
6) Repeat the following steps for each of the 6 segmented images.
   a. Repeat the following steps for each segment.
      i. Build a 3D colour histogram (3DRGB) for the segment and normalize the (4×4×4 = 64) feature vector.
      ii. Find the LBP feature vectors which belong to the segment and compute the Euclidean distance between these feature vectors and each centre.
      iii. Find the minimum distance between the LBP feature vectors and each centre.

   iv. Build the 64-element Bag of Words Feature Vector (BoWFV) by counting the frequencies of the best matching code-word (the minimum distances).

b. Build a K-d tree for the BoWFVs of all segments and find the 2nd Approximate Nearest Neighbours for each element in the tree.

c. Find the neighbouring BoWFV which satisfy:

   i. The Euclidean distances between the BoWFVs is less than$\mathrm{Threshold}_1$.

   ii. The Euclidean distance between the 3DRGB histograms is less than$\mathrm{Threshold}_2$.

d. Build a binary image (MaskImage) for all neighbouring segments that satisfy i & ii above.

7) To produce a more robust result, sum the six binary images and recolour all the pixels which have a value $>=$ 2. These pixels have been detected as a potential CMF in two different superpixel scales.

5.2.2 The Experiment to detect CMF using SLIC superpixel segmentation

The SLIC algorithm [7] divides the image into irregular blocks which exhibit state-of-the-art boundary adherence. Therefore, we used SLIC to segment the copy-move objects and maintain its boundaries synchronously.

The size of duplicated objects can vary from one image to another, and appear as small or large parts of the image. The required number of approximately similar-sized superpixels (K), is the parameter that controls the SLIC [7] algorithm. Experimentally, we found that K has a significant influence on segmenting the copy-move objects, because selecting the correct K will segment the duplicated objects in approximately the same way. Six possible numbers of similar-sized superpixels have been determined experimentally; K= [30, 55, 100, 300, 550, 800].

The Local Binary Pattern [113] has been used to describe each segment texture, and a 3D colour histogram has been used to describe the colour distribution of each segment.

The LBP is one of the popular methods of describing the image texture and we have used it to describe the texture of the segment. We built the LBP image, where each LBP label value is represented as a pixel, and computed the 16×16 LBP histogram for each pixel in that image.

The LBP feature vectors which belong to each segment have been found and we want to define each segment by one feature vector only. We used the Bag of Visual Words [114] technique and the LBP operator to describe the texture of each segment. We determined 64 centres in the LBP feature vectors as a whole and computed the distance between each feature vector belonging to a segment and the centres. Then we bin the minimum distance to build the Bag of Visual Words LBP feature vector for each segment.

The colour distribution was found for each segment using a 3 Dimension RGB histogram (4×4×4=64 feature elements).

We built the K-d tree [115] for the Bag of Visual Words LBP feature vectors and found the 2nd Approximate Nearest Neighbours for each feature vector. We used the Fast Library for Approximate Nearest Neighbour (FLANN) [96], which can significantly reduce the time required for searching. We determined the neighbouring vectors (segments) that had a Euclidean distance less than Threshold_1 (0.1) and had a difference between its colour distribution that was less than Threshold_2 (0.05). We generated a black and white mask for the primary matched segments and then summed the 6 generated masks since one mask is generated for each of the different scales of the segments. Then recoloured the pixels which had a value equal to or greater than two.

Below are two examples of the proposed algorithm in use and the results that were generated, see Figure 5-1 and Figure 5-2.

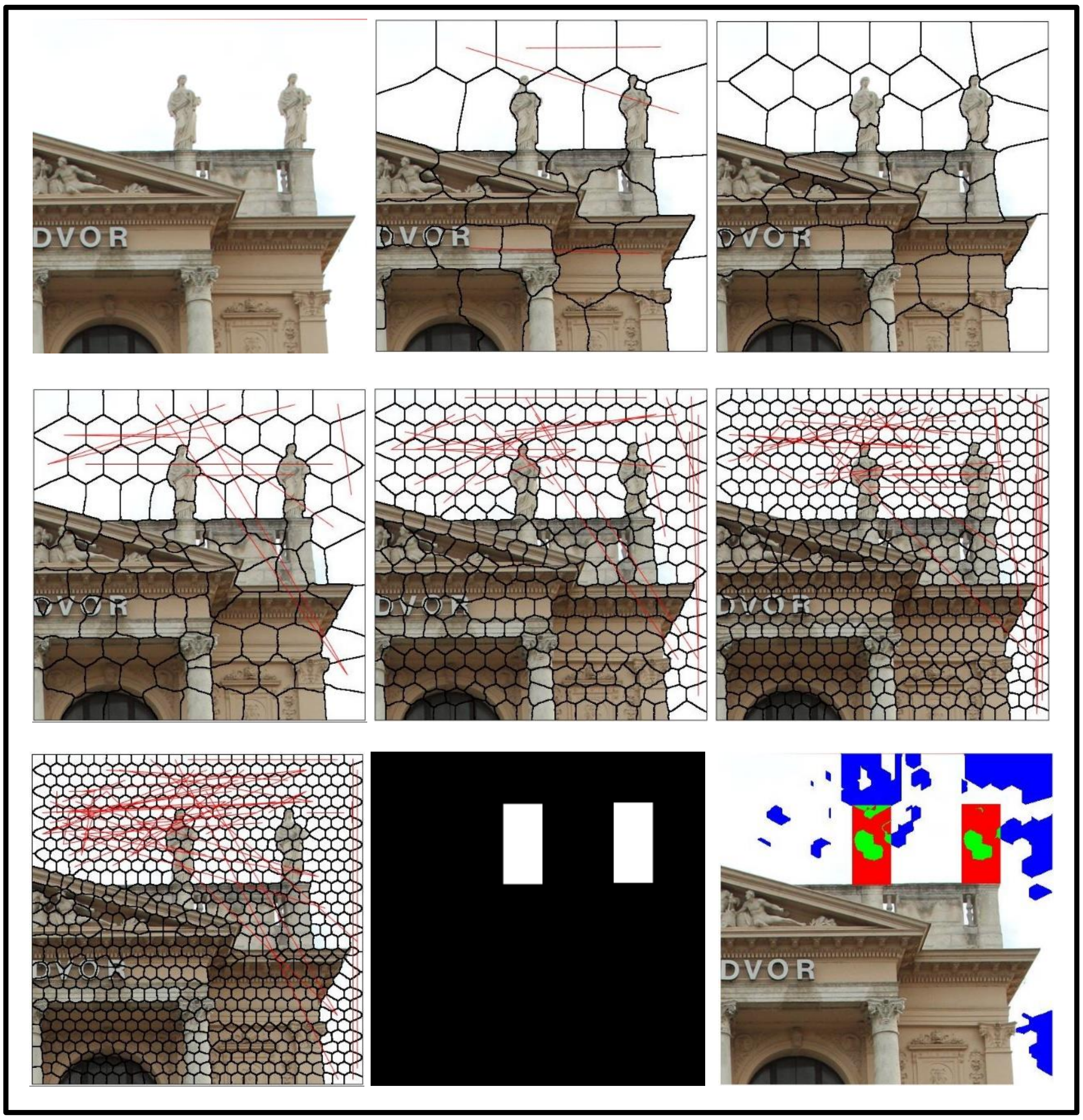

Figure 5-1: (top to bottom, left to right) Input Forged Image, segment image A using six different scales and the matched areas, the ground truth and final result.

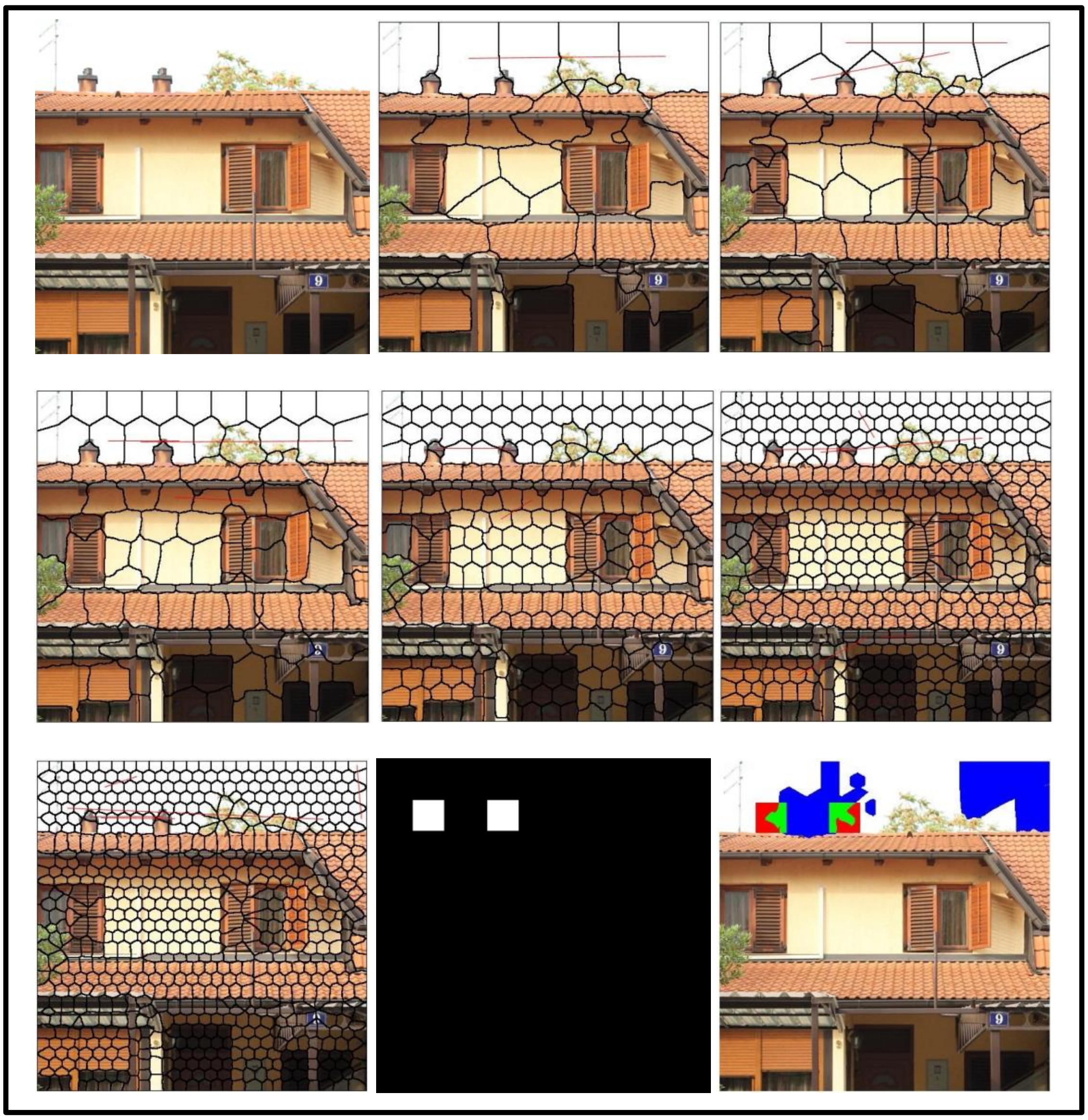

Figure 5-2: (top to bottom, left to right) Input Forged Image, segment image B using six different scales and the matched areas, the ground truth and the final result.

5.2.3 Analysis of the proposed algorithm to detect CMF using SLIC superpixel segmentation

The duplicated object is moved from one location to another in the same image. Therefore, the pixels adjoining the copy-move object are different from the pixels that were neighbours of the original object.

Experimentally, we found that the SLIC [7] cannot segment the copy-move objects exactly consistently, but it can segment the copy-move objects in a similar way, depending on the required size of the segment itself. The number of required segments (K), which controls the size of the segment in the SLIC, the colour and the spatial proximity is used in SLIC segmentation.

Our experiments found this approach to be unreliable (F-measure = 0.146).

The main reasons for the algorithm failure are as follows:

a) The SLIC cannot segment the copy-move objects exactly consistently, see Figure 5-3 and Figure 5-4.

b) The LBP histogram and 3DRGB histogram are densely extracted from each segment. In consequence, the segmentation result has a significant influence on the feature vectors that are produced. Experimentally, we found that the discriminability of these features is too sensitive to the inconsistency in the segmentation of the copy-move objects.

Although the suggested method can partially detect the forgery in some images, the final detected image contains a high percentage of false matches. Moreover, in many images the suggested method cannot detect forgery at all. The proposed method is also time-consuming: the average time required to process one image is about 27 minutes.

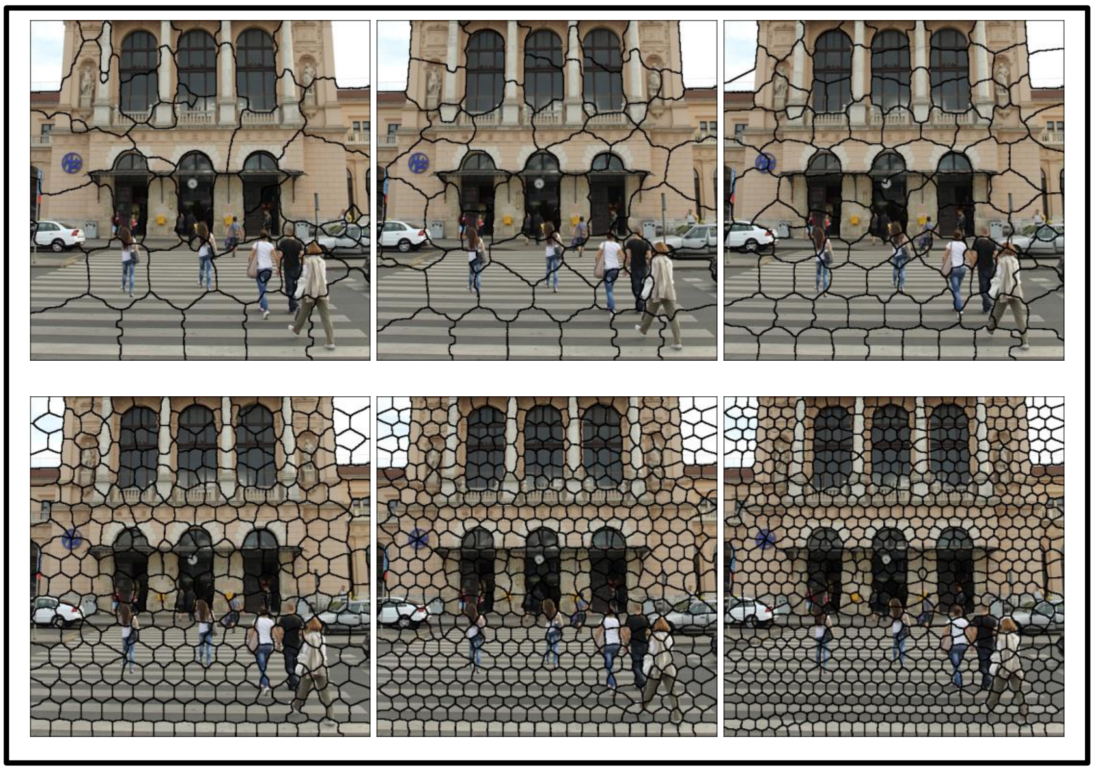
Figure 5-3: Segment an image using six different scales.

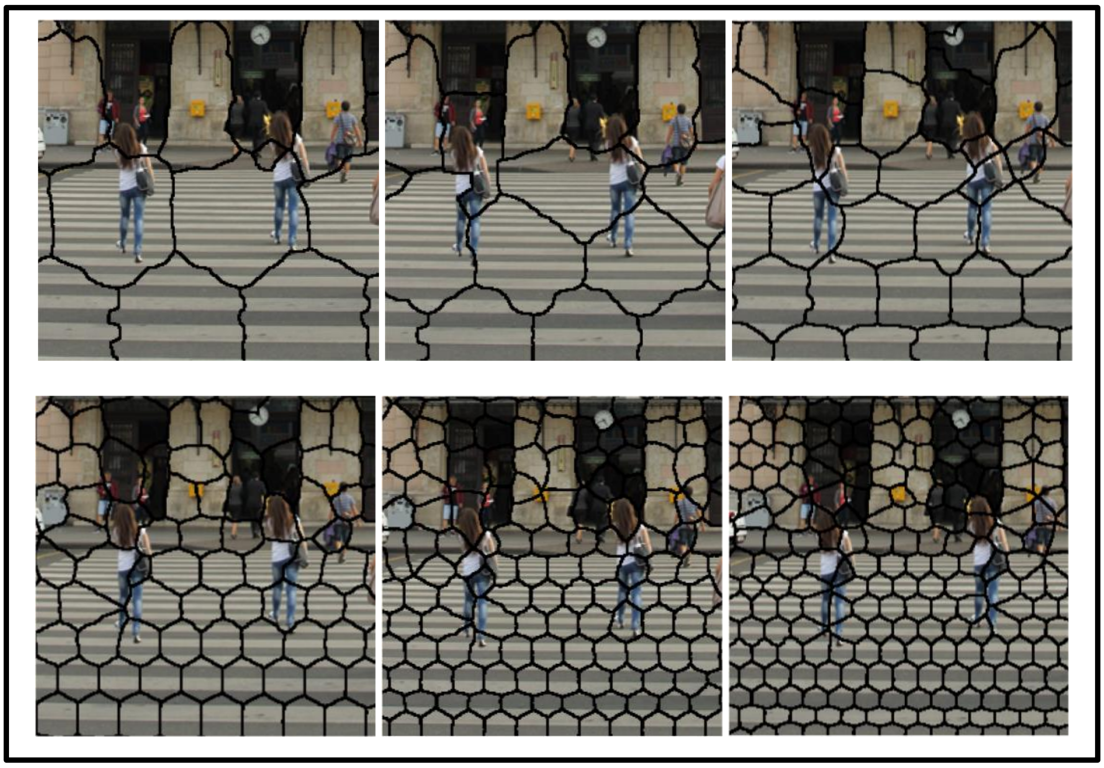
Figure 5-4: Zoomed CMF areas on segmented image using six different scales.

Other methods [81][27] have used segmentation to divide the image into small irregular regions and sparsely extract the SIFT features from each segment. As discussed above, this approach is similar to locating matching features between different images using SIFT, since each segment can be treated as a different image. Given that the segmentation methods have been to divide the original image into various patches (small images), the approach above depends on extracting SIFT features at the level of segments rather than from the whole image.

5.3 Using Bag of Words Image (BoWImage) to detect Copy-Move Forgery

We proposed a new segmentation method using the LBP and the Bag of Visual Words. First, we generated the LBP image by computing the LBP label for each pixel in the image and assigned this label to the centre pixel. We described each 16×16 overlapped patch in the LBP image using a 256-histogram feature vector. The K-means++ was used to find 16 centres in the feature vectors. The number of required centres has been optimized experimentally, and we found using 16 centres can segment the CMF objects in the most acceptable way.

We computed the Euclidean distances between each feature vector and the K centres and found the minimum distance. The minimum distance indicates which centre the feature belongs to and we used this centre number to label the centre pixel of each block.

For each different value of K (16), we remove the small segments. Then the HOG [86] and the Hu Moments [116] have been used to describe the shape and the appearance of each segment. We extracted the HOG from each pixel belonging to a particular segment and computed the mean for these features. Also, we extracted the Hu moments from each segment and concatenated the HOG, the Hu moments and the size of the segments in one feature vector. We found the matched segments/feature vectors within specific thresholds. Finally, cascade methods were used to remove the outliers.

5.3.1 High-level description of the proposed algorithm for detecting Copy-Move Forgery using Bag of Words Image (BoWImage)

This is a high-level description of our proposed algorithm for detecting CMF using BoWImage segmentation:

1) Convert the image to grayscale if it is coloured.
2) Compute the Local Binary Pattern (LBP) for each pixel in the grayscale image.
3) Histogram each 16×16 patch in the LBP image to build a 256-element LBP Feature Vector (LBPFV) and save the coordinates of the centre pixel.
4) Use the K-means++ to find 16 centres (K=16) in the LBP feature vectors.
5) Generate a blank Bag of Words Image (BoWImage).
6) Repeat the following steps for each LBP Feature Vector.
    a. Compute the Euclidean distances between the LBP Feature Vector and each centre and find the minimum distance.
    b. Assign the centre label with the minimum distance to the Bag of Words image (BoWimage) in the corresponding coordinates.
7) Repeat the following steps for each label:

Find all the segments (i.e. the connected components with the same label) and compute the size of each segment.

Repeat the following for each segment

a. Remove small segments less than Threshold_1 in size.

b. Compute the (16×16) Histogram of Oriented Gradients (HOG) for each pixel in the segment.

c. Compute the mean of HOG feature vectors to generate a single feature vector for each segment (HOGFV).

d. Compute the Hu moments for each segment and normalize the seven-element feature vectors (HuFV).

e. Concatenate the HuFV, HOGFV and the area of each segment.

f. Save the centroid and the bounding-box for each segment.

g. Build a K-d tree for the concatenated feature vectors and find the 2nd Approximate Nearest Neighbours for each element in the tree.

h. Find the neighbouring segments which satisfy:

   I. The Euclidean distance between the concatenated feature vectors is less than Threshold_2.

   II. The difference in size of the neighbouring segments is less than Threshold_3.

i. Save the centroid and the bounding-box for the two matched segments.

This method generates a good deal of false matching, which we eliminate by using the cascade approach:

1) **Remove the false matching using the centroids and HOG**

```
Input: List A, List B, where $A_i$ and $B_i$ are the centroids for two matched segments.
Output: List C and D, where $C_j$ matches $D_j$,
Repeat the following for each element in the list.
    If the Euclidean distance between HOGs for each centroid in Lists A and B is less than $Threshold_4$
        a. Mark the centroids of $A_i$, and $B_i$ as matched.
        b. Find the index of ($A_i$, $B_i$) in the list of centroids and use this index to save the bounding-box in $C_j$ and $D_j$ respectively.
    End if
End repeat
```

2) **Remove the false matching using the Bounding-Box and HOG (pixel level)**

```
Input: List C of Bounding-Box, List D of Bounding-Box, where $C_j$ matches $D_j$.
Output: List1 and List2, where $List1_i$ matches $List2_i$.
    1) Read the Bounding-Box for each matched segment where its centroids have also been matched.
    2) If the size of the two Bounding-Boxes is not equal, randomly select pixels from the larger box equal to the
```

size of the smaller box. This will make the size of the two bounding-boxes similar.

a. Build a K-d tree for the HOG feature vectors which extracted from the bounding-boxes.

b. Find the 2nd Approximate Nearest Neighbours for each element in the K-d tree.

c. Find the matched pixels which have the Euclidean distance between their HOG feature vectors less than Threshold_5.

3) Apply RANSAC on List1 and List2.

5.3.2 Experiment to detect Copy-Move Forgery using Bag of Words Image (BoWImage)

The LBP value (label) was computed for each pixel in the image and assigned to the centre pixel to produce the LBP image. Then each 16×16 overlapped patch in the LBP image was described by a histogram which produces a 256-feature vector for each patch. K-means++ clustering [117] has been used to find K centres in the feature vectors. We tested different numbers of clusters (8, 16, 24 and 32) and found that K=16 produces better results than any other. Then the Euclidean distances were computed between each feature vector and the K centres.

Each cluster was considered as a visual word that represented a particular local pattern common to the feature vectors in the cluster. Then, similar feature vectors were clustered together, such that the minimum Euclidean distance between each feature vector and the K centres indicated which centre the feature belonged to. Then we used this centre number to label the centre pixel of each block.

The generated Bag of Words image has the number of values equal to the number of centres (K=16), for it will be recalled that this number was optimized experimentally. We found that the Bag of Words Image can cluster (segment) the copy-move objects in an almost similar way. This inspired us to use this method to detect copy-move forgery.

Therefore, we inferred that the potential copy-move objects, or part of them, belonged to the same value and were clustered in almost the same way.

For each different value, the Connected Component Labelling method [118] was used to extract a set of geometric properties (e.g. area, centroid, bounding-box, etc.) from each segment. Then all the segments less than a particular threshold in size were removed.

The HOG [86] and the Hu Moments [116] have been used in many applications of object detection to describe the shape and the appearance of the objects. Therefore we used the HOG and Hu moments in our suggested algorithm to describe the shapes of each segment in their different values.

The HOG was extracted from each pixel belonging to a particular segment. Then, to represent each segment by one feature vector, the mean was computed for all the HOG features that were extracted from a particular segment. The Hu moments were also used to describe the shape of each segment as its translation, rotation and scaling invariant. Then the HOG, Hu moments and the area of each segment were concatenated to produce the (44=36+7+1)-element feature vector.

A K-d tree has been built to all the feature vectors of a specific value, and the 2nd Approximate Nearest Neighbour has been found for each segment (feature vector). We determined which of the neighbouring segments had a

Euclidean distance between the concatenated feature vectors of less than a threshold ($\mathrm{Threshold}_2$=7) and where the difference between the size of a segment and that of its neighbours was less than a threshold ($\mathrm{Threshold}_3$=2000). The colour of the two matched segments has been changed to green, and the centroids and the bounding box also have been saved.

The proposed algorithm produces a number of false matches, which we reduced as far as possible by cascading false match removal algorithms. First, the HOG was extracted from the centroid of each matched segment. The Bounding-Boxes of the matched segments were saved when the Euclidean distance between the HOG of the centroids was less than a threshold ($\mathrm{Threshold}_4$= 0.07).

Second, the saved Bounding-Box for each two matched segments was used to reduce the false matches. When the sizes of the Bounding-Boxes of the two matched segments were not equal, the smaller one was stretched to equal the size of the larger one of the matched segment. Then the HOG was extracted from each pixel of the two matched Bounding-Boxes (segments). A K-d tree was built for the feature vectors of the two segments and the 2nd ANN was found for each element in the binary tree. The coordinates of the two neighbours were saved whenever the Euclidean distance between its feature vectors was less than a threshold ($\mathrm{Threshold}_5$= 0.03). All thresholds were experimentally optimized to produce the best possible results.

Then RANSAC was applied on the two generated lists to eliminate any possible false matches.

The experimental results illustrate that the performance of this suggested algorithm (F-measure = 0.467) is better than that of our previous proposed algorithm, which used SLIC superpixel (F-measure = 0.146).

Below are two examples of using the proposed algorithm and the generated results, see Figure 5-3 and Figure 5-4.

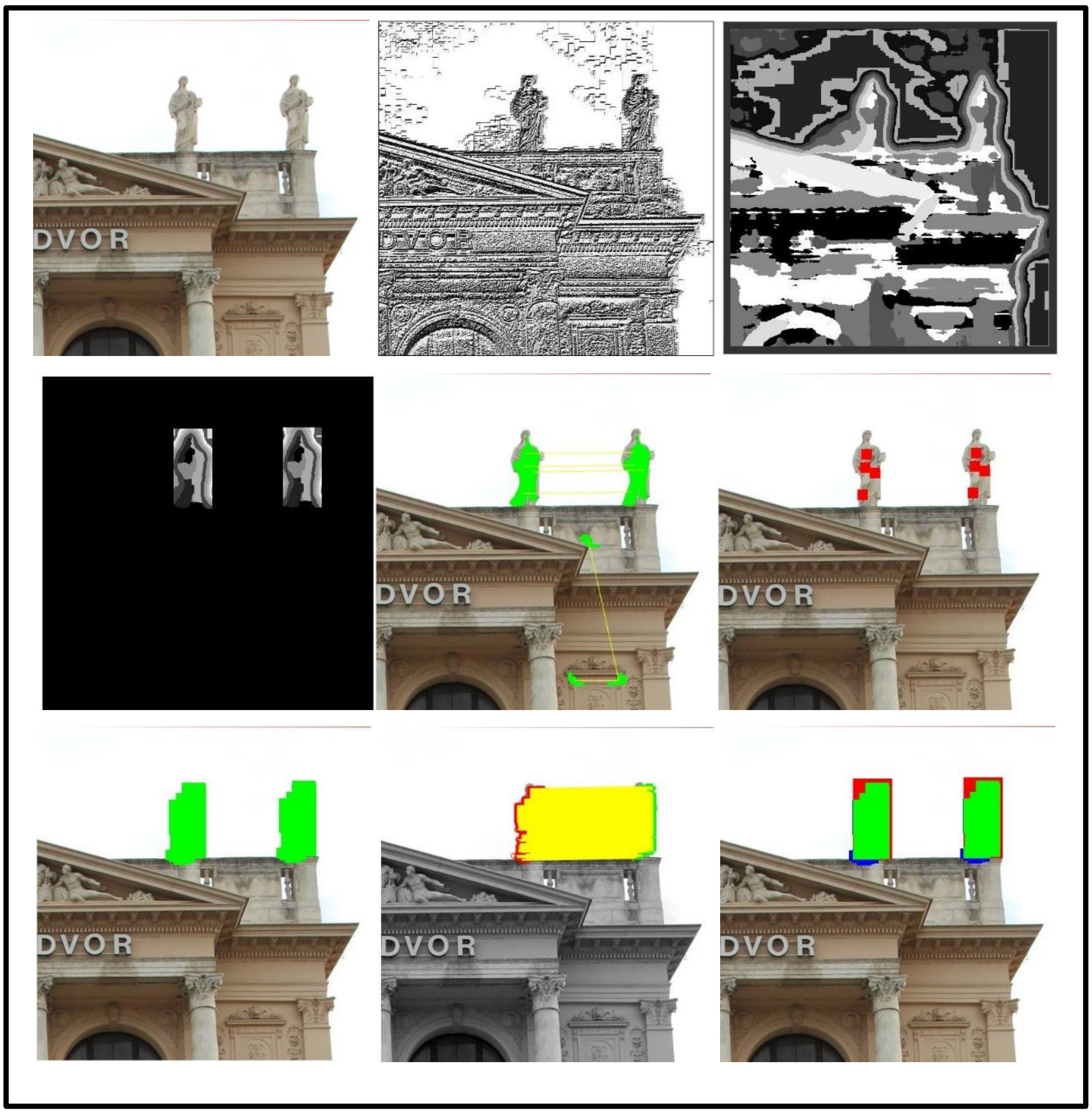

Figure 5-5: (top to bottom, left to right): Input forged image , LBP code image, Bag of Words Image (BoWImage), Ground truth-BoWImage, Initial matched segments, Matched centroid Image, Matched Bounding-Box image, RANSAC image and final result.

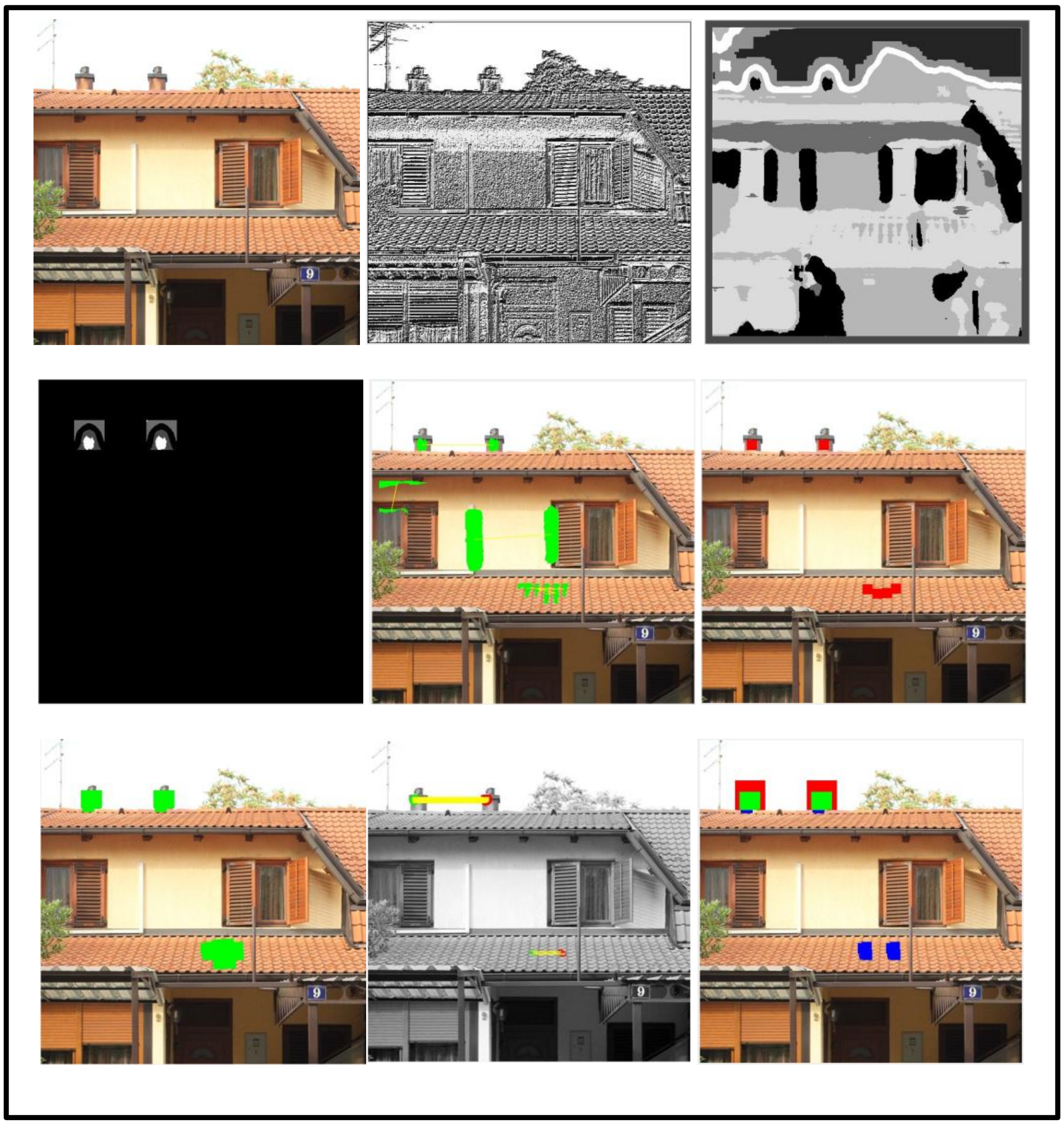

Figure 5-6: (top to bottom, left to right): Input forged image, LBP code image, Bag of Words Image (BoWImage), Ground truth-BoWImage, Initial matched segments, Matched centroid Image, Matched Bounding-Box image, RANSAC image and final result.

5.3.3 Analysis of the proposed algorithm to detect CMF using Bag of Words Image (BoWImage)

The main reasons for the failure of this algorithm are as follows:

a. The Bag of Words cannot segment the copy-move objects consistently, because the surrounding area of the copy-move objects sometimes affects the result of the clustering.

b. The segmentation errors have a significant influence on the generated feature vectors because the features are extracted densely from each segment. These changes between the feature vectors of the duplicated objects produce the mismatches (False Negative), see Figure 5-8.

c. Alternatively, the segmentation errors produces wrong matches (False Positive), especially in the flat regions and similar textures. The proposed cascade algorithms to remove the false matches can partially reduce the FP areas, but it cannot eliminate them, see Figure 5-7.

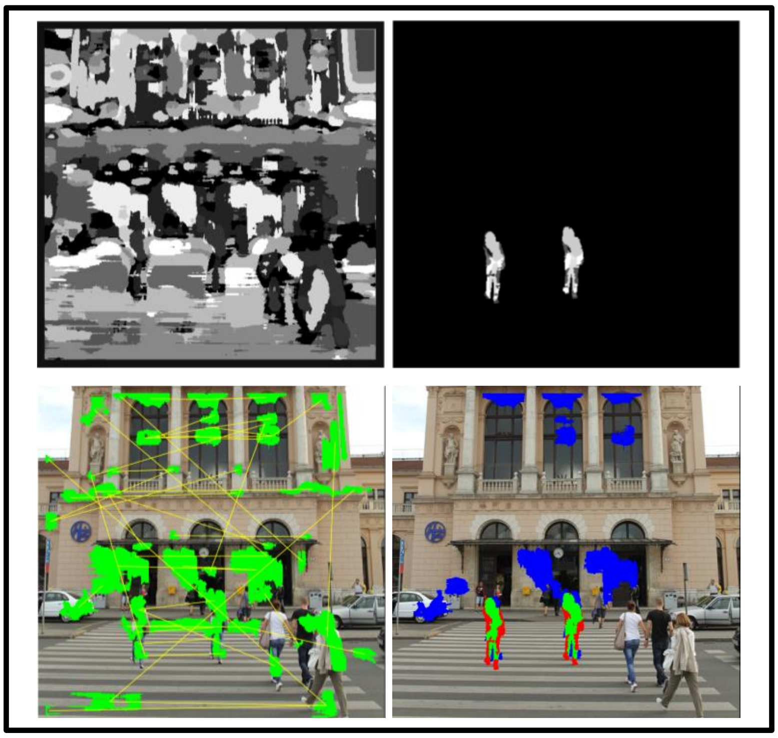

Figure 5-7: Bag of Words Image (BoWImage) of image, Ground truth-BoWImage, Initial matched segments and final result.

The F-measure is quite low, illustrating the unreliability of the BoWImage algorithm. Moreover, this method is time consuming: the average time required to process one image is about 15 minutes.

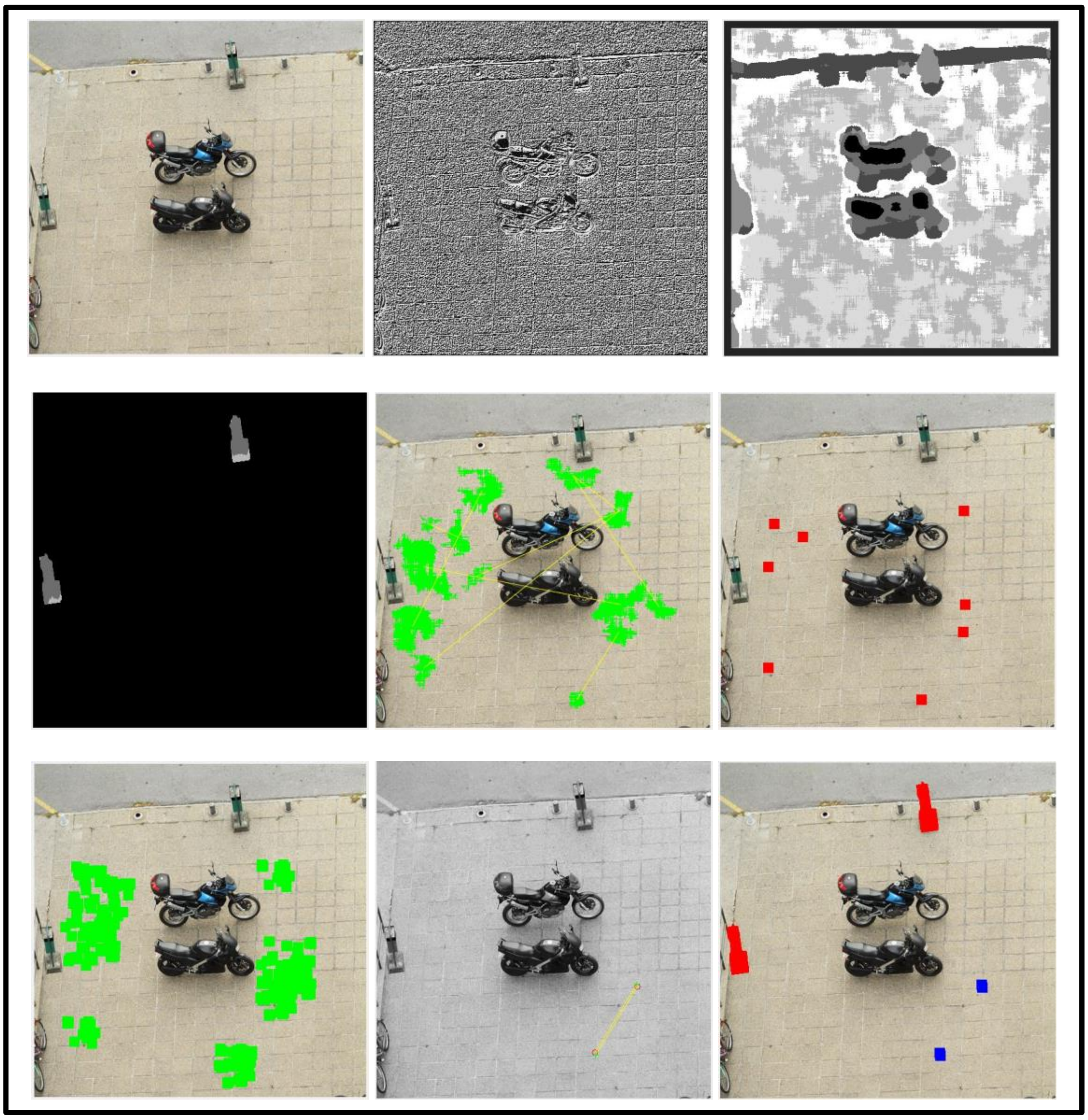

Figure 5-8: (top to bottom, left to right): Input forged image, LBP code image, Bag of Words Image (BoWImage), Ground truth-BoWImage, Initial matched segments, Matched centroid Image, Matched Bounding-Box image, RANSAC image and final result.

5.4 Using Rolling Guidance Filter and Multi-thresholding to Detect Copy-Move Forgery

5.4.1 A Threshold Selection Method using Gray-Level Histograms

The most common and simplest method of segmenting an image is to use thresholding. Otsu [119] suggested a method to find the best threshold to binarize the grayscale image, see Figure 5-9.

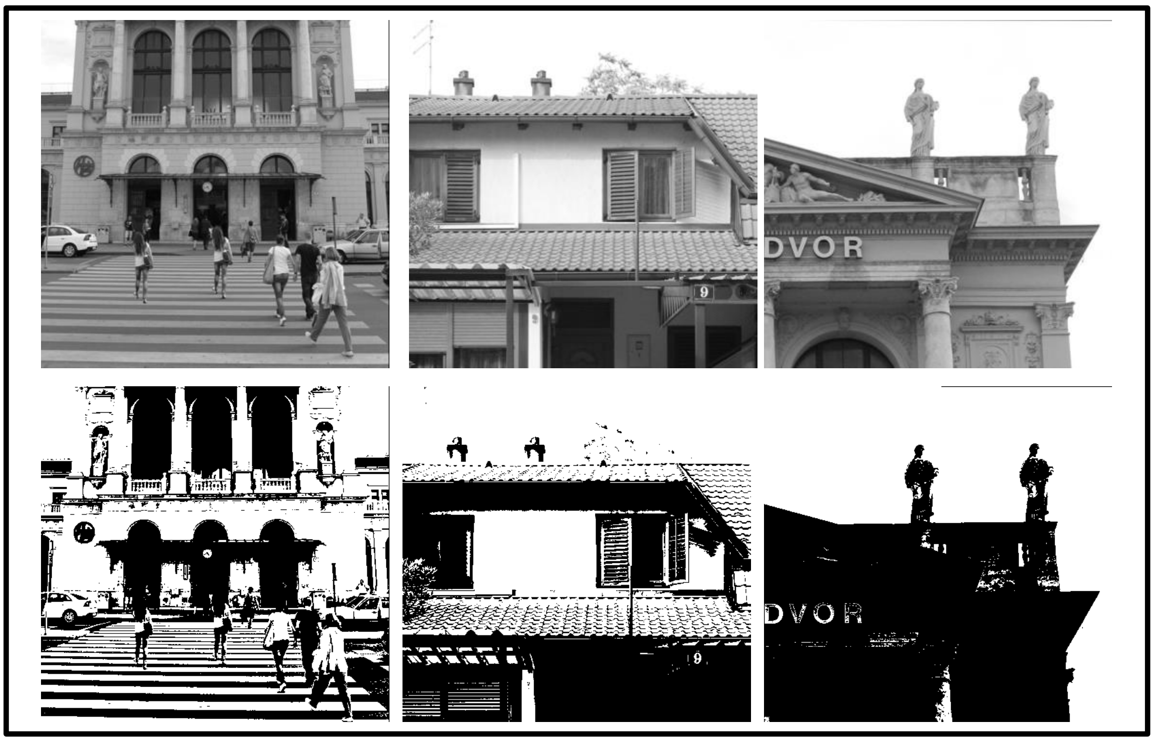

Figure 5-9: an example of using Otsu's method to convert a grayscale image to a binary image.

As shown in Figure 5-9, the binary images which are generated from applying a single Otsu threshold on the grayscale images contain many small segments, and these would be unstable for matching in CMF detection. Moreover, using the multi-level thresholding version of Otsu [120] does not produce a better result, see Figure 5-10 and Figure 5-11.

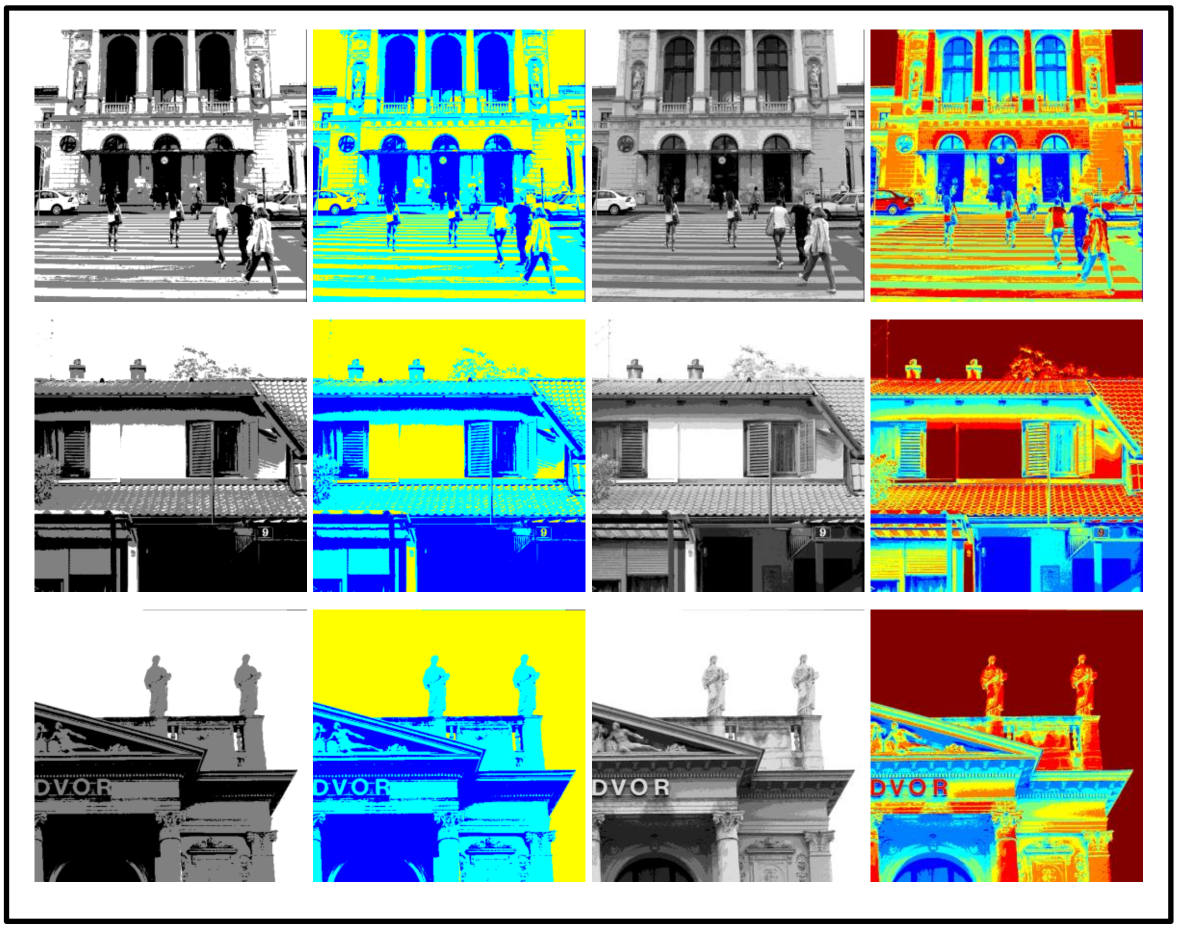

Figure 5-10: (each row, left to right) Segmented image using 2 thresholds, Segmented image using 2 thresholds with RGB labelling, Segmented image using 7 thresholds, Segmented image using 7 thresholds with RGB labelling .

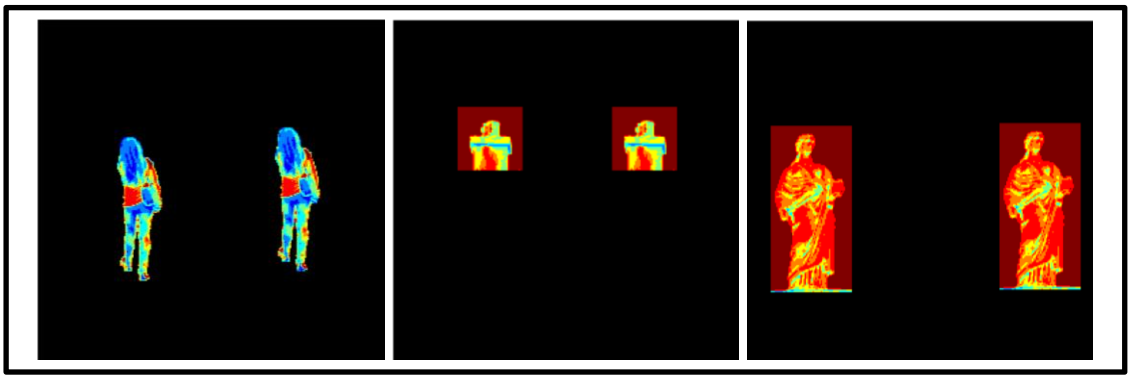

Figure 5-11: Zoomed CMF areas.

5.4.2 Rolling Guidance Filter

In order to produce more meaningful segments, we filtered the image to remove noise and unnecessary details. The images contain significant structures and edges over a range of scales [121]. The structure scale can be defined as the smallest Gaussian standard deviation σ_s, so that when this σ_s Gaussian deviation is applied to an image the corresponding structure disappears.

However, it is not possible to use the Gaussian filter as a scale-aware filter because it blurs all edges and cannot distinguish between strong and weak edges. In other words, the Gaussian average mechanism removes the edges of structures smaller than the smoothing scale, but it also blurs large-scale structures instead of removing them.

However, many filtering techniques have been proposed to smooth the image while maintaining these structures. Edge-aware filters have been used to remove the low-contrast edges (gradual changes) and preserve the high-contrast edges, e.g. the bilateral filter [122], guided filter [123] and weighted median filter [121].

We used the Rolling Guidance Filter [121] because it has been shown to be effective for removing small-scale structures while preserving large-scale structures by the use of scale-aware local operations. It can cope with irregular shapes and furthermore has low computation cost, see Figure 5-12.

The two main steps in the rolling guidance method:

1. **Remove small structures:** In this step, the Gaussian filter is used. However, as well as removing the edges of structures smaller than the smoothing scale, it also blurs large-scale structures instead of preserving them.

2. **Edge recovery:** A joint bilateral filtering of the given input image and the image from the previous iteration is used to recover the edges. This can be understood as a filter that smooths the input image, guided by the structure of the previous iteration image.

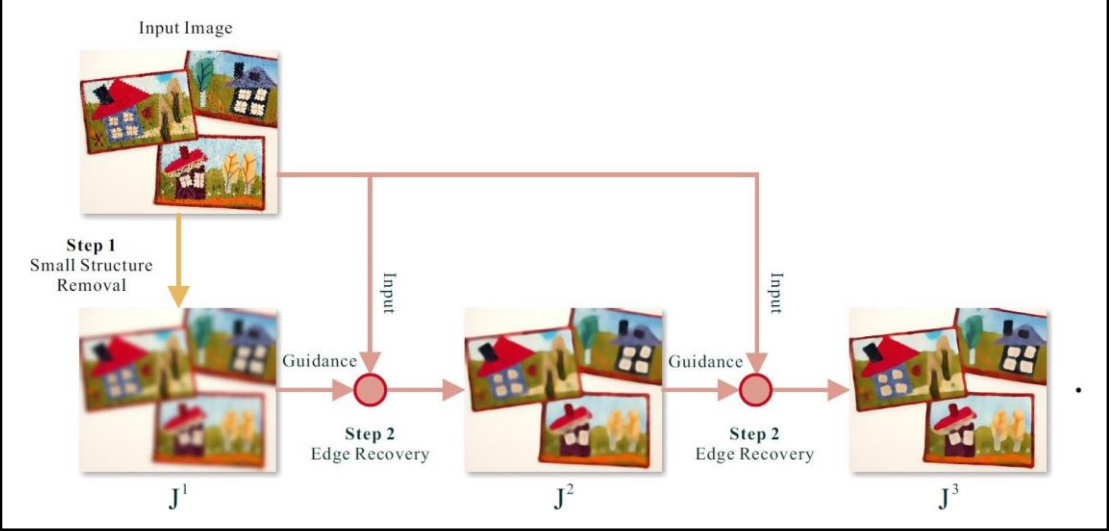

Figure 5-12: Flow chart of the guiding rolling method [121].

The binary image which is generated from applying the single Otsu threshold on the Rolling Guidance filtered image (smoothed image) produces a reasonably segmented image. As shown in the binary image of Figure 5-13, the detrimental or unwanted content has been removed and the pixels have been clustered appropriately. However, this does not adequately segment the copy-move objects in the image. Under-segmentation has caused the objects to become merged with the background.

Therefore, we used the multi-level thresholding version of Otsu applied after the Rolling Guidance filtering, Figure 5-14 and 5-15.

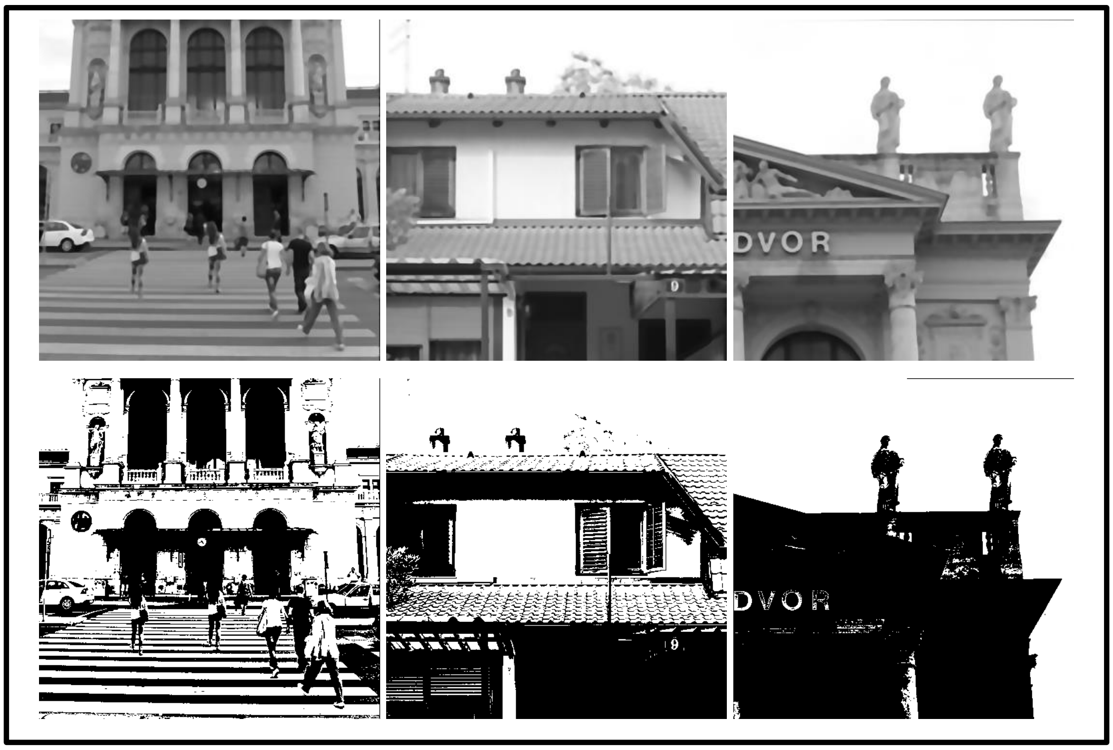

Figure 5-13: an example of using the Otsu method on the Rolling Guidance filtered image to convert a grayscale image to a binary image.

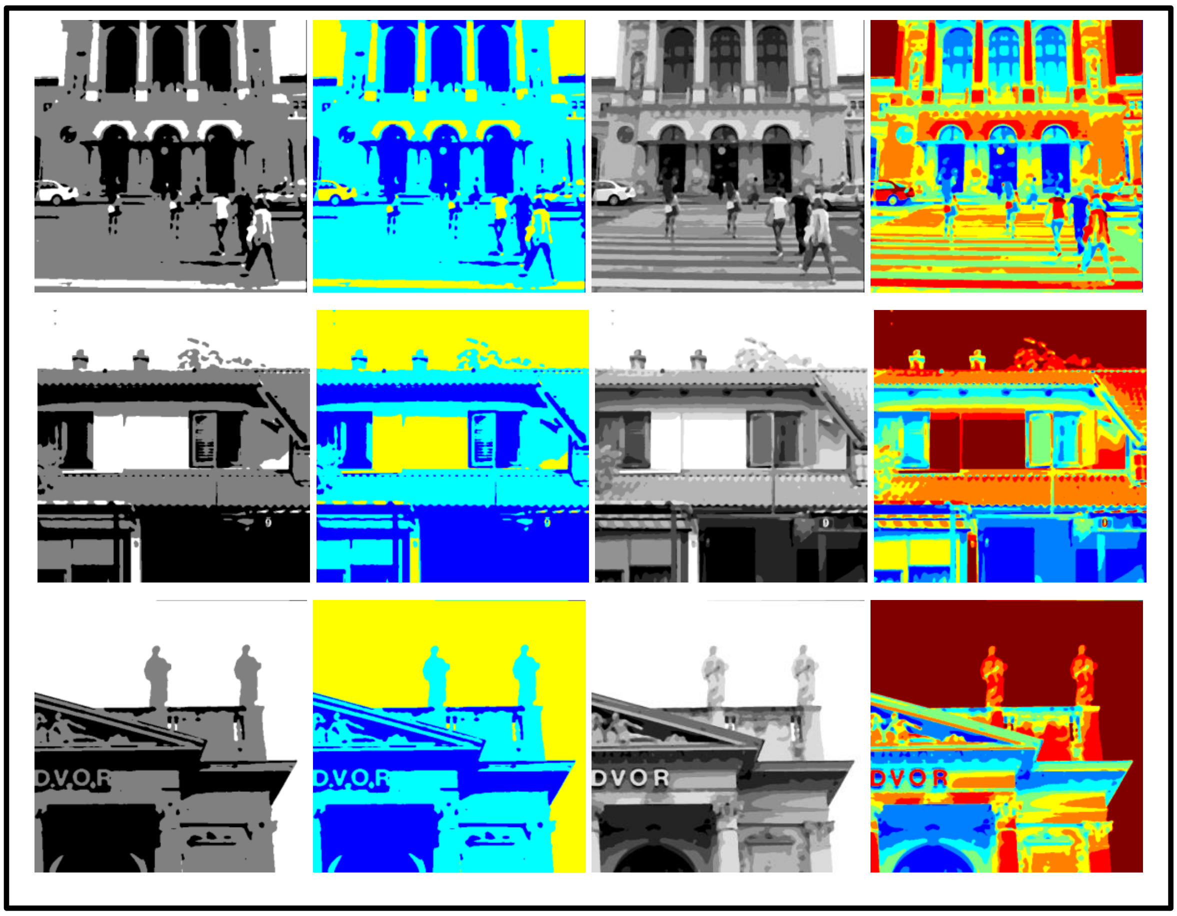

Figure 5-14: (each row, left to right) Segmented RGImage using 2 thresholds, Segmented RGImage using 2 thresholds with RGB labelling, Segmented RGImage using 7 thresholds, Segmented RGImage using 7 thresholds with RGB labelling .

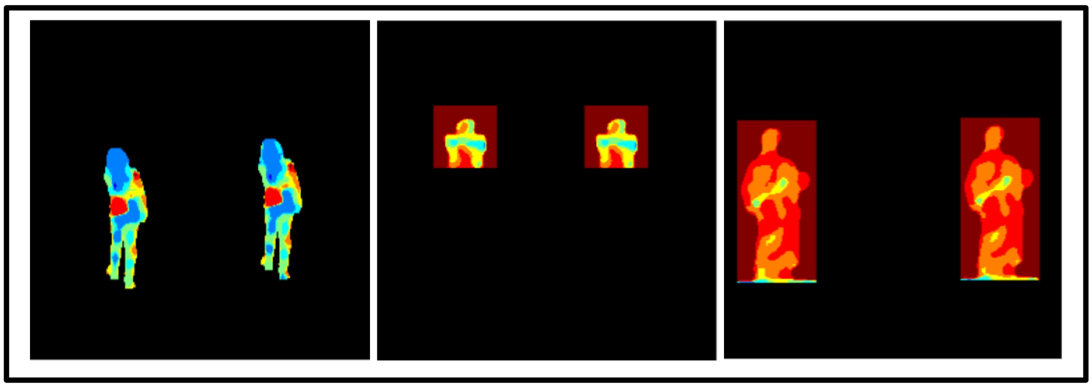

Figure 5-15: Zoomed CMF areas in RGImage.

5.4.3 High-level description of the proposed algorithm to detect CMF using the Rolling Guidance Filter and Multi-thresholding

We used the Rolling Guidance filter [121] to smooth the image and preserve the strong edges, then we used the Otsu method [119] to find 7 different thresholds on the filtered image. The 7 thresholds were used to quantize the Rolling Guidance filtered image into 8 different labels and the small segments were removed. Then for each segment, we built the 3DRGB histogram and the Segment Gradient Orientation Histogram (SGOH), which is described in section 5.4.4.

We built a K-d tree to all the feature vectors (segment) of a specific label and found the 2ANN. We determined the potential CMF segments (neighbouring segments) whose Euclidean distance between their concatenated feature vectors was less than a specifics threshold and the difference between their sizes was not significant. Finally, we used RANSAC to remove the outliers and the hysteresis technique to grow the detected copy move regions.

This is the high-level description of our algorithm for detecting CMF using Rolling Guidance filtering and multi-threshold segmentation:

1) Filter the coloured image using the Rolling Guidance method to generate a Rolling Guidance Image (RGImage).
2) Use the Otsu method to find 7 different thresholds on the RGImage.
3) Quantize the RGImage using the 7 thresholds.
4) Repeat the following steps for each different label (colour).

a. Compute a set of properties for each connected component object (segment).

b. Remove small segments which are less than Threshold_1 in size.

c. Repeat the following for each segment

    i. Find the pixels' coordinates and the area of each segment.

    ii. Build a 3D colour histogram (3DRGB) for each segment and normalize the $(4\times4\times4 = 64)$-feature vector.

    iii. Build the SGOH feature vector, see Section 5.4.4.

d. Build a K-d tree for the concatenated feature vectors and find the 2nd Approximate Nearest Neighbours for each element in the tree.

e. Find the matched segments (a binary image of the primary detection) which satisfy:

    I. The Euclidean distance between the concatenated feature vectors less than Threshold_2.

    II. The difference in the size of the matched segments is less than Threshold_3.

f. Save the coordinates of the matched segments in ListA and ListB.

5) Make the size of ListA equal to the size of ListB:

- For each matched segment

```
            ▪ If size (ListA(i)) is not equal to size (ListB(i))
                  Find the larger segment and randomly select
                  pixels from it equal to the size of the smaller
                  segment.
            ▪ End if
      • End for
6) Apply RANSAC on ListA and ListB.
7) Use hysteresis to grow regions of ListA and ListB.
8) Mark all matched segments.
```

5.4.4 The algorithm proposed for building a Segment Gradient Orientation Histogram (SGOH)

This algorithm is inspired by the SIFT context [3], but SIFT uses a fixed block size (16×16), and we cannot use it to describe irregular shapes (segments).

To make the SGOH rotation invariant, we used the moment method to find the canonical orientation for each segments. Then we rotated each segment to the estimated ordination and build an 18 bins gradient magnitude weighted orientation histogram. Finally, the feature vector was normalised between one and zero.

1. Use the moment method (intensity centroid measure) [93] to find the canonical orientation for each segment.
2. Rotate each segment according to its canonical orientation to make the descriptor rotation invariant.

3. Construct a gradient magnitude weighted orientation histogram containing $18 = 360°/20°$ bins.
4. Normalize the generated Segment Gradient Orientation Histogram (SGOH) feature vector between zero and one [102].

5.4.5 Experiment to detect Copy-Move Forgery using a Rolling Guidance Filter and Multi-thresholding

The Rolling Guidance filter [121] was used to smooth the image and maintain the strong edges, then the Otsu method [119] was used to find 7 different thresholds on the filtered image. We tried different numbers of thresholds (5, 7, 9, 11 and 13), and experimentally found that using 7 thresholds can consistently segment the copy-move objects much better than using any other number of thresholds. These 7 thresholds were then used to quantize the Rolling Guidance filtered image into 8 different labels. Next, connected component labelling was used to find the number of objects (segments) in each different value (thresholds) and to compute a set of properties (e.g. area, pixel list, etc.) for each object. Then all the segments smaller than 50 pixels (Threshold_1=50) were removed, recalling that this threshold had been optimized experimentally.

A 3D colour (3DRGB) histogram was used to describe the colour distribution of each segment and a Segment Gradient Orientation Histogram (SGOH) was built to represent the gradient of each segment. The steps for building our SGOH descriptor were described in Section 5.4.4.

A K-d tree was built to all the feature vectors of a specific label and the 2nd Approximate Nearest Neighbour was found for each segment (feature vector). The neighbouring segments were determined whose Euclidean distance between their concatenated feature vectors was less than a threshold (Threshold_2= 0.012) and the difference between their sizes was less than a threshold (Threshold_3=1.5). We evaluated the effect by varying Threshold_2 in the range 0.001 to 0.05 in increasing powers of 0.003 on a 20% sample of the images. Below 0.005 we found too few matches and above 0.01 too many, in the range 0.01 to 0.02 there was little impact on the results, so we selected the value 0.012.

The size of the two matched segments was made equal, save their coordinates in two separate lists and RANSAC applied to remove the outliers. Finally, use the hysteresis technique to grow the detected copy move regions and re-colour them green.

5.4.5.1 CMF detection with translation

The experimental work illustrates that the performance of the suggested algorithm is much better than our previous proposed segmentation algorithms, as the F-measure=0.79. The proposed algorithm successfully detected plain duplication (translation) on 36 out of 40 images, see Figure 5-16, Figure 5-17 and Figure 5-18 . Moreover, this method is less complicated and much faster than the previous suggested methods; one image can be processed in about 50 seconds.

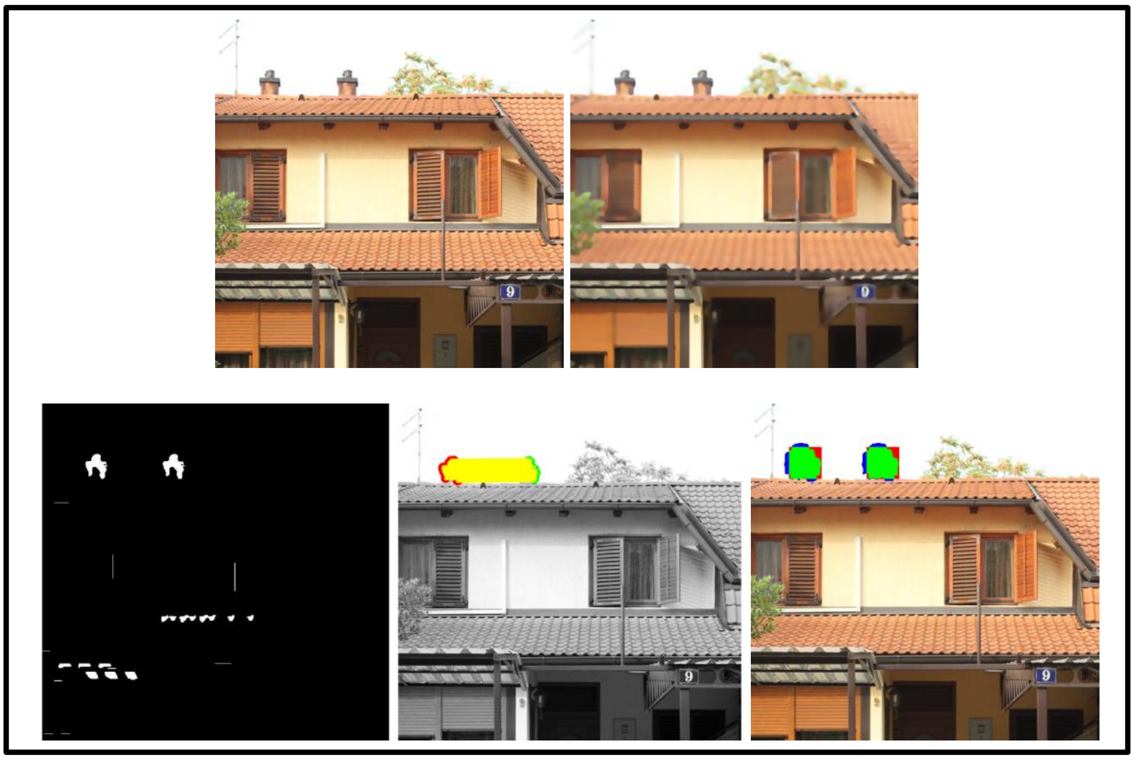

Figure 5-16: (top to bottom, left to right) input forged image, Rolling Guidance image, primary detection, RANSAC result and final result after hysteresis.

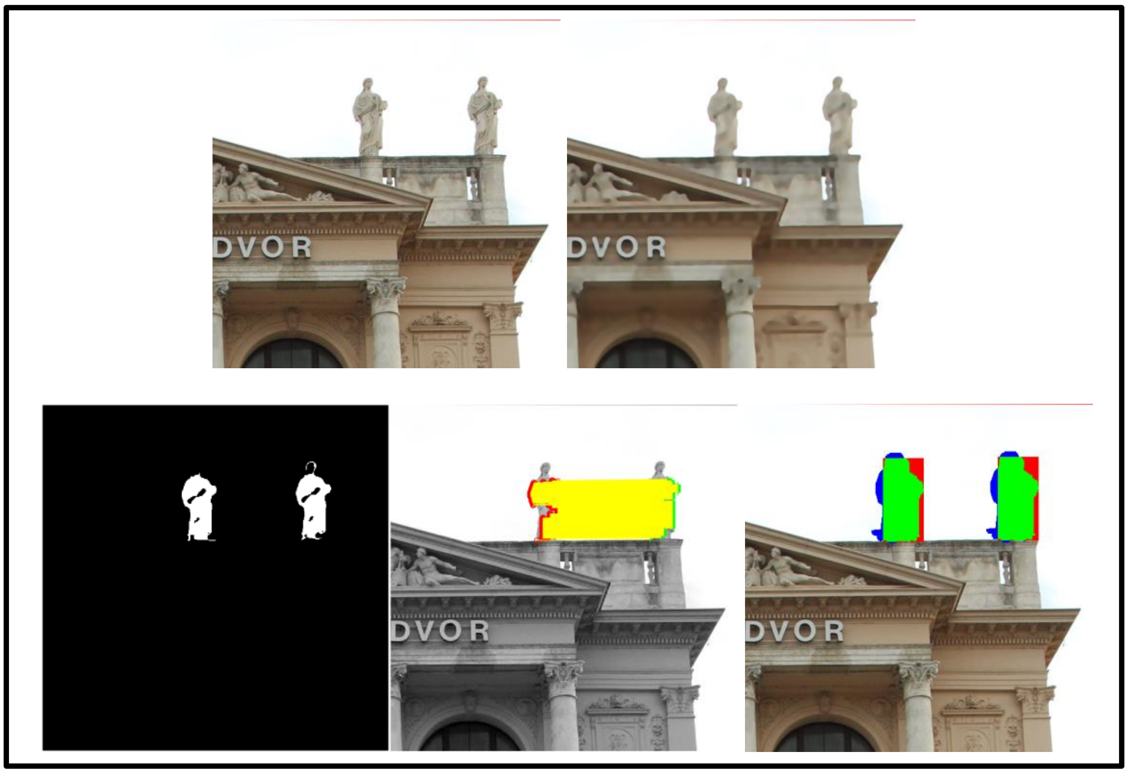

Figure 5-17: (top to bottom, left to right) input forged image, Rolling Guidance image, primary detection, RANSAC result and final result after hysteresis.

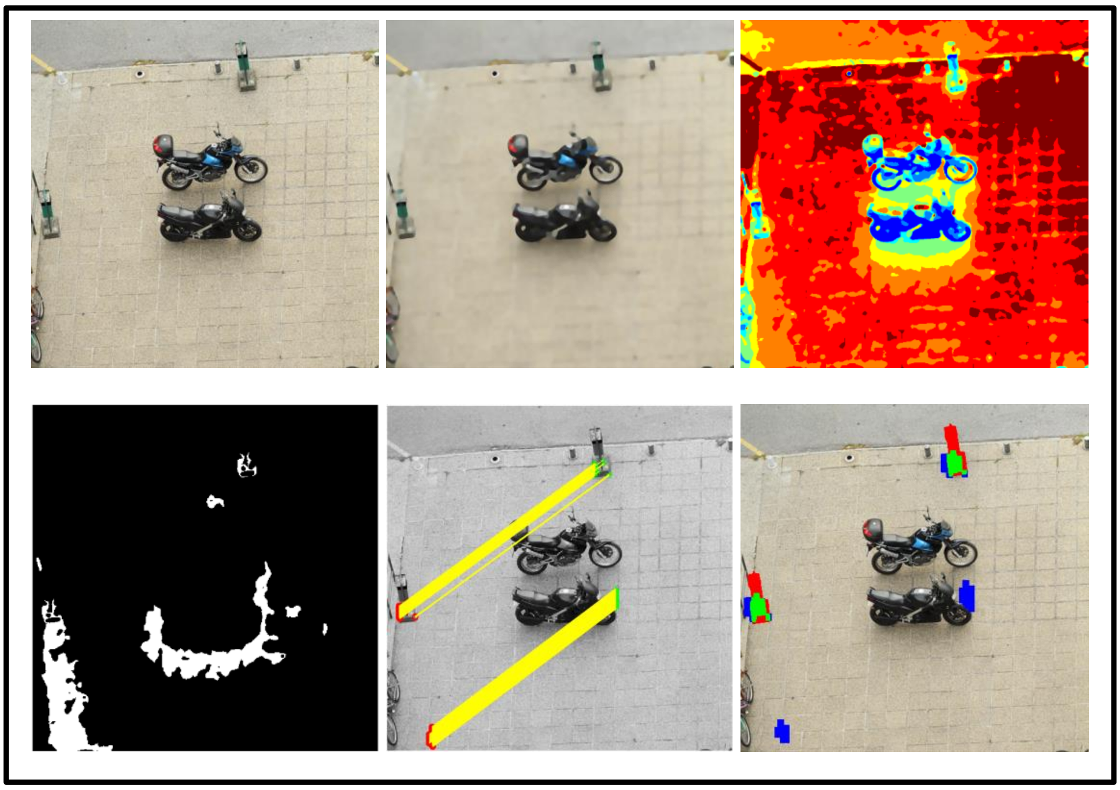

Figure 5-18: (top to bottom, left to right) input forged image, Rolling Guidance image, Quantized image primary detection, RANSAC result and final result after hysteresis.

5.4.5.2 CMF detection with translation and post-Processing (attacks)

In our work, we considered different types of attack (image blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise), see Figure 5-19 – Figure 5-24 and Table 5-1.

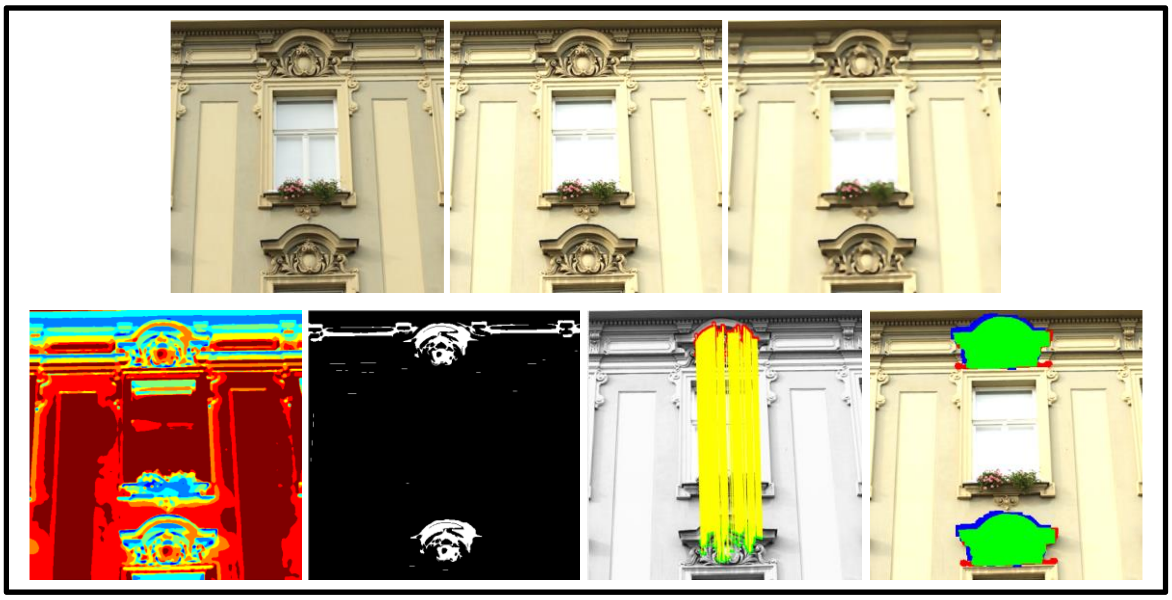

Figure 5-19: (top to bottom, left to right) Input forged image, Forged image after Brightness Change, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

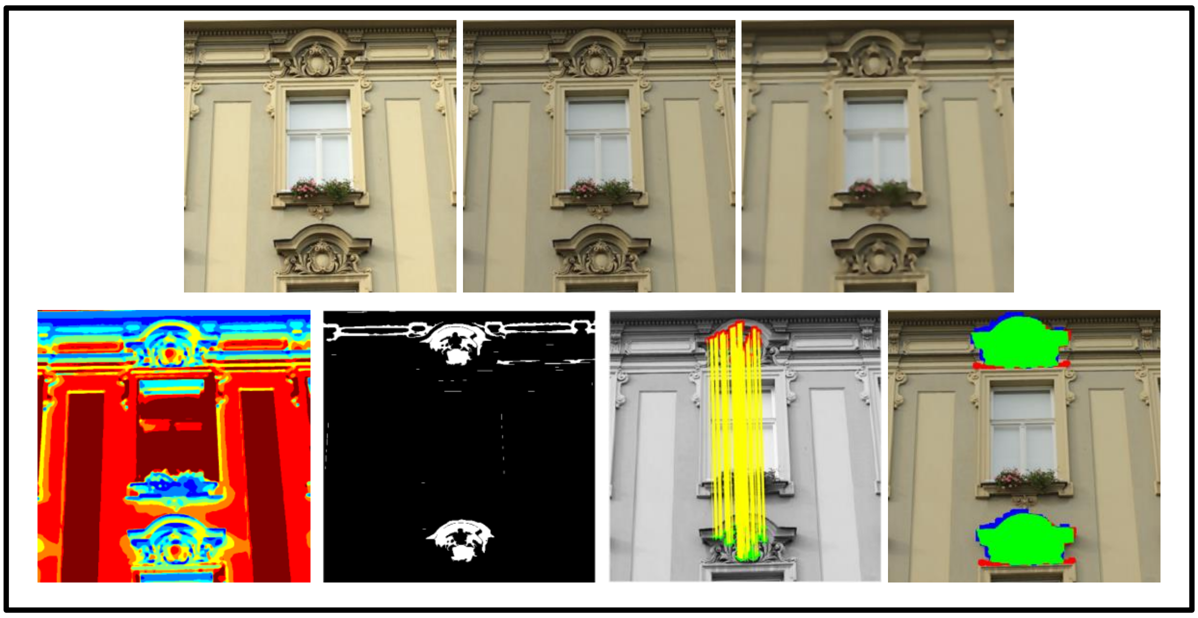

Figure 5-20: (top to bottom, left to right) Input forged image, Forged image after Contrast Adjustment, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

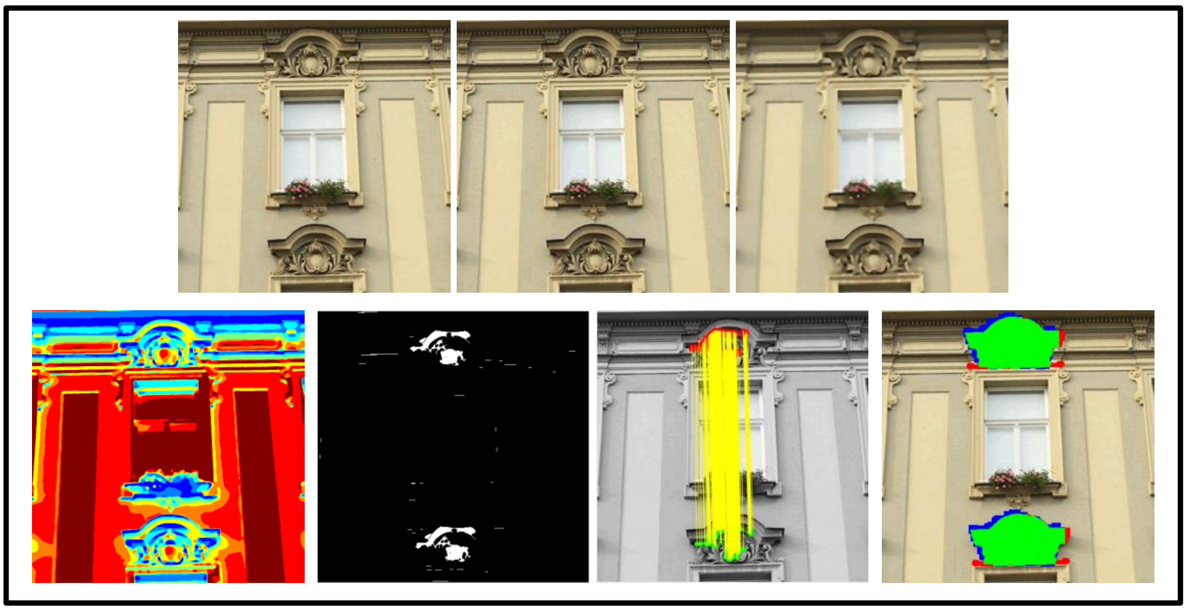

Figure 5-21: (top to bottom, left to right) Input forged image, Forged image after Colour Reduction, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

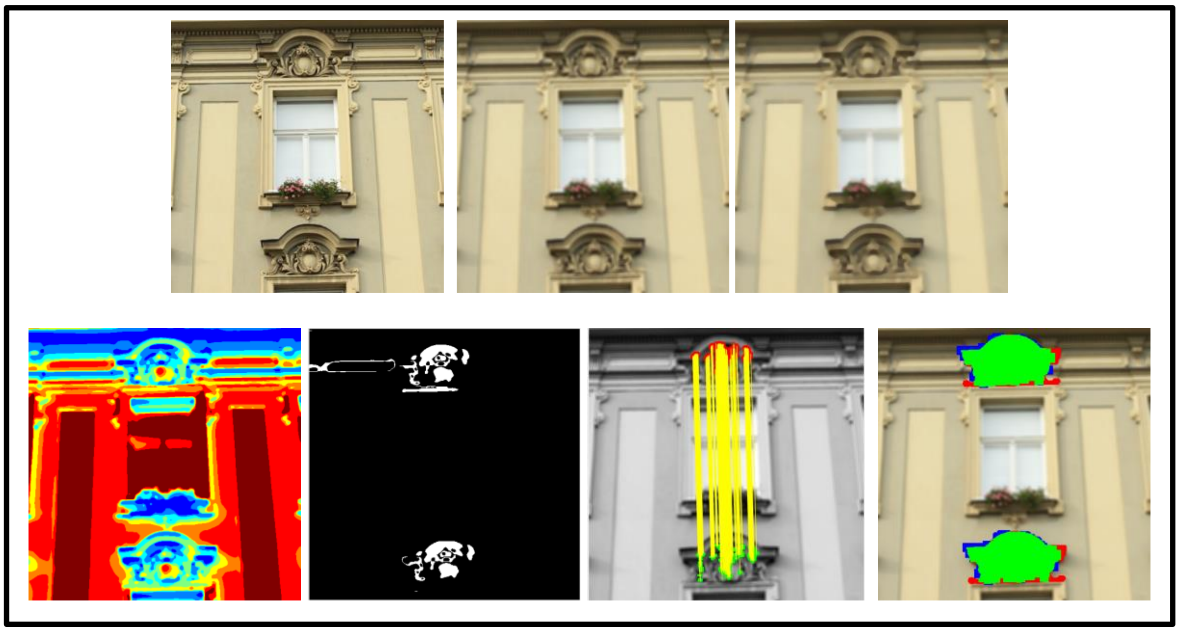

Figure 5-22: (top to bottom, left to right) Input forged image, Forged image after image blurring, Rolling Guidance image, Quantized image, Primary detection, RANSAC result, and final result after hysteresis.

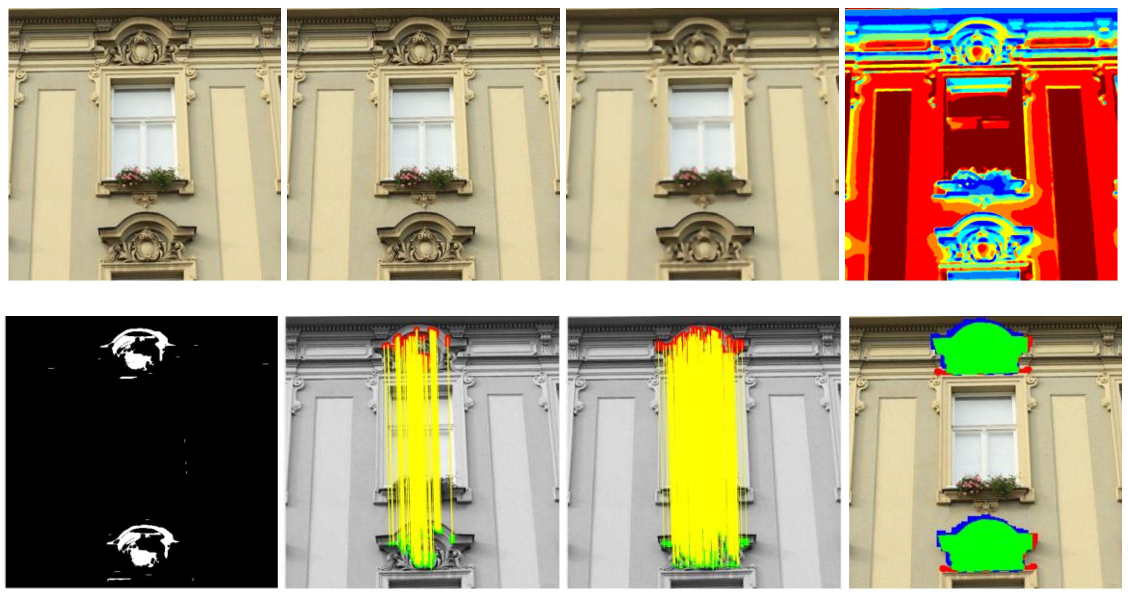

Figure 5-23: (top to bottom, left to right) Input forged image, Forged image after adding noise, Rolling Guidance image, Quantized image, Primary detection, RANSAC result, Hysteresis 5th iteration and final result after hysteresis.

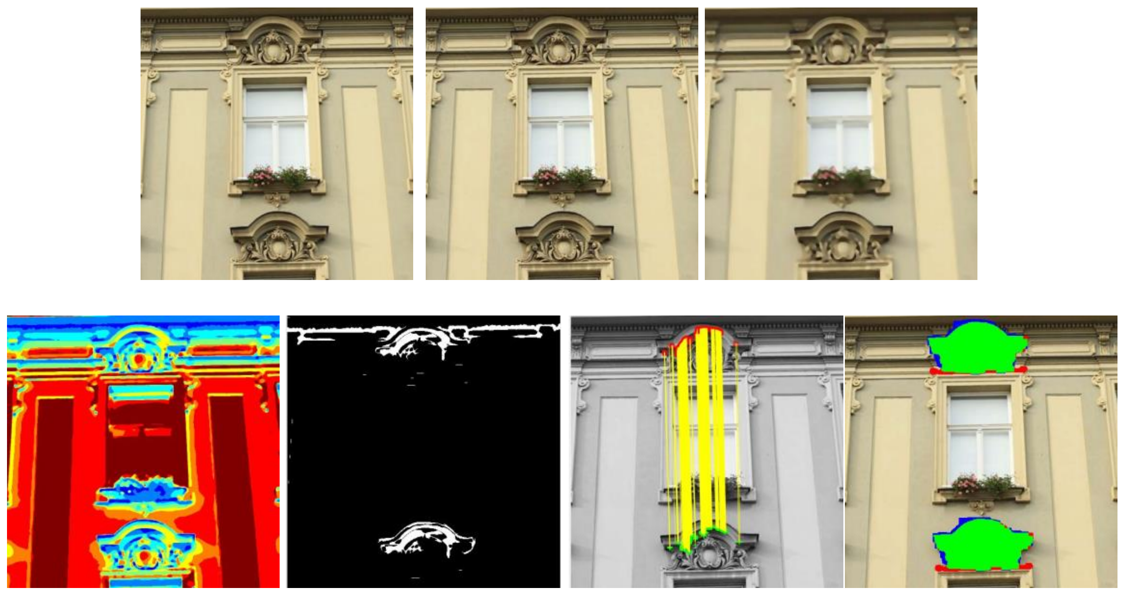

Figure 5-24: (top to bottom, left to right) Input forged image, Forged image after JPEG Compression, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

Table 5-1 CMF detection with translation and post-processing (Attacks)

Post-processing	F-measure	Number of Detected images out of 40
Translation without attack	0.79	36
Brightness Change Range(0.01, 0.8)	0.78	36
Contrast Adjustment Range (0.01,0.8)	0.78	36
Colour Reduction (32 intensity levels)	0.78	36
Image Blurring (5×5 average filter)	0.79	36
Adding Noise (μ=0,σ^2=0.0005)	0.69	28
JPG Compression (quality factor=40)	0.78	36

5.4.5.3 CRMF detection (rotation)

Using rotation invariant features is the primary requirement of copy-rotate-move forgery detection. The Segment Gradient Orientation Histogram (SGOH) is rotation invariant, as each segment is rotated to its canonical orientation before computing the weighted histogram, see Section 5.4.4. The experimental work illustrates that the suggested algorithm can detect rotated duplicated objects to an acceptable standard. The algorithm detected forgery on 35 images out of 40 with F-measure=0.71, see Figure 5-25, Figure 5-26, Figure 5-27 and Table 5-2.

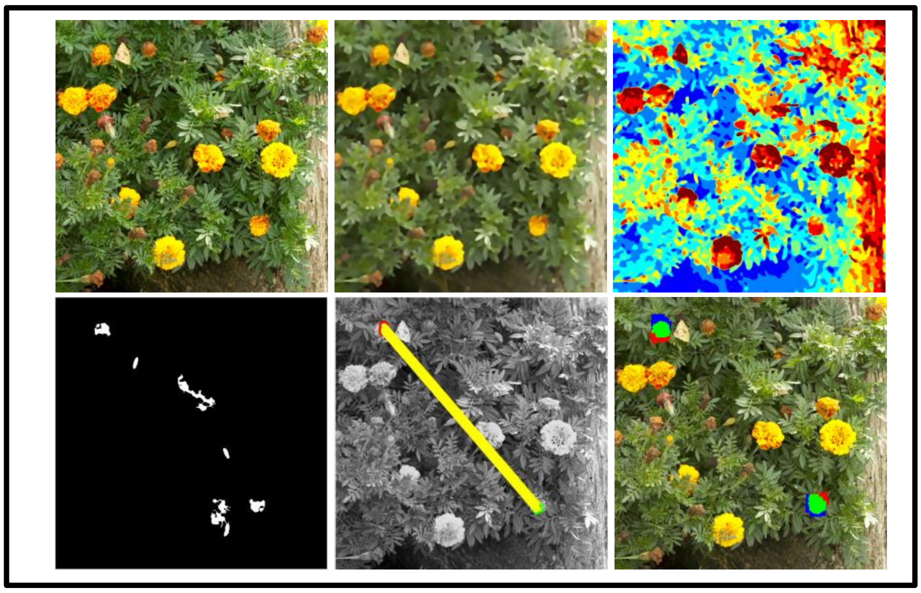

Figure 5-25: (top to bottom, left to right) Input forged image with copied object rotated by 180°, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

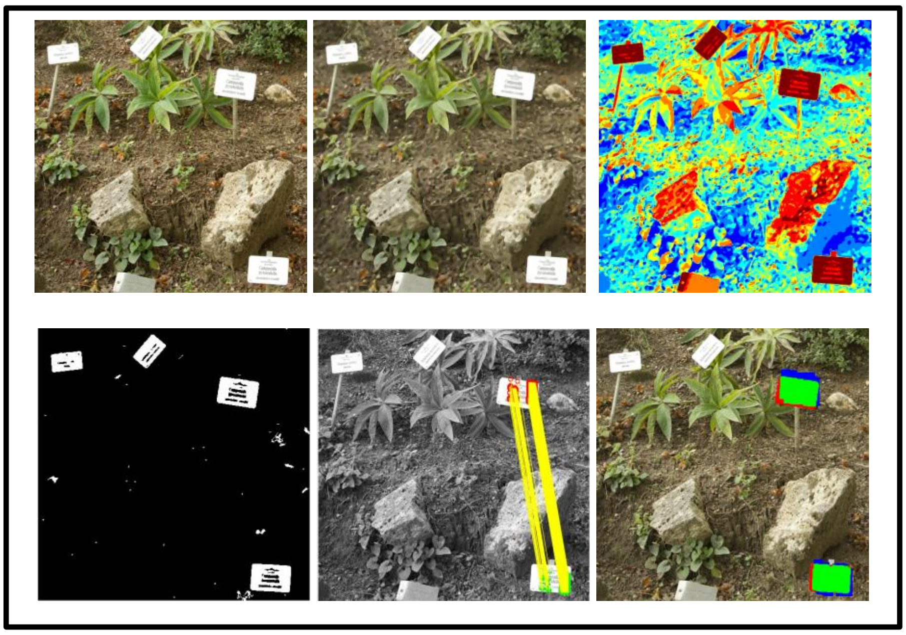
Figure 5-26: (top to bottom, left to right) Input forged image with copied object rotated by 5°, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

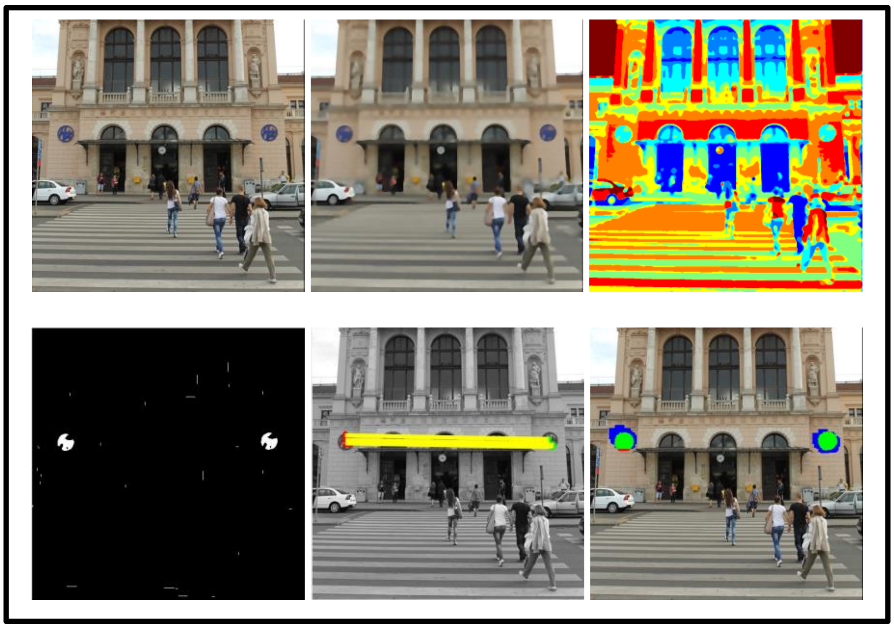
Figure 5-27: (top to bottom, left to right) Input forged image with copied object rotated by -3°, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

5.4.5.4 CRMF detection with post-Processing (attacks)

Different types of attack (image blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise) were considered in our work, see the figures below and Table 5-2.

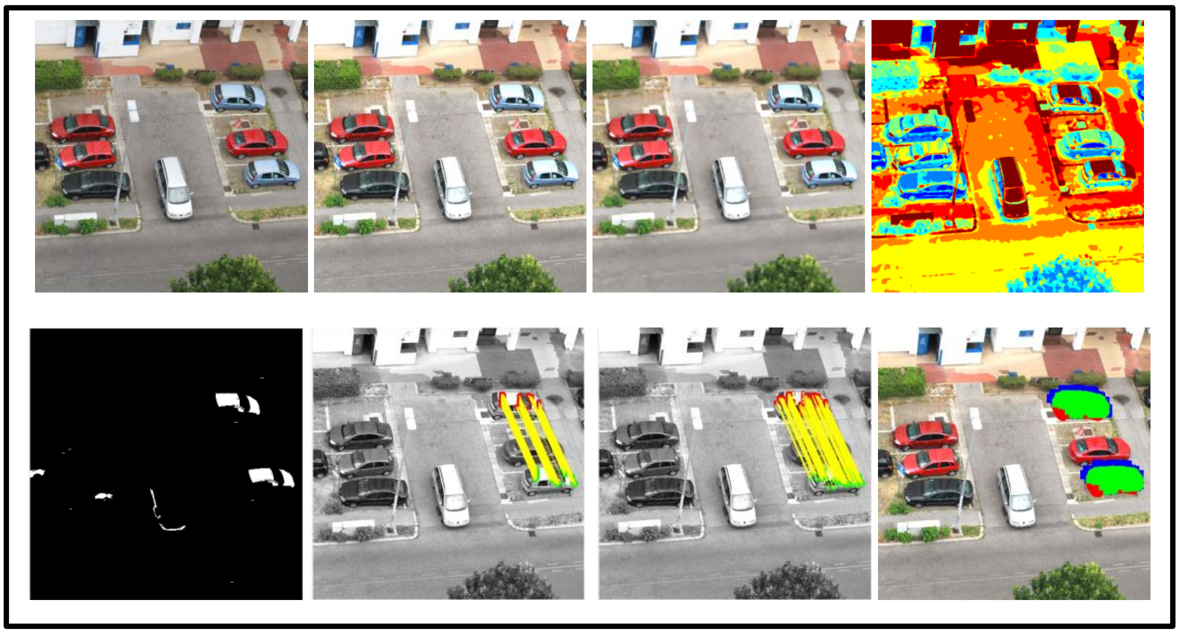

Figure 5-28: (top to bottom, left to right) Input forged image with copied object rotated by 5°, Forged image after Brightness changing, Rolling Guidance image, Quantized image, Primary detection, RANSAC result, Hysteresis 22nd iteration and final result after hysteresis.

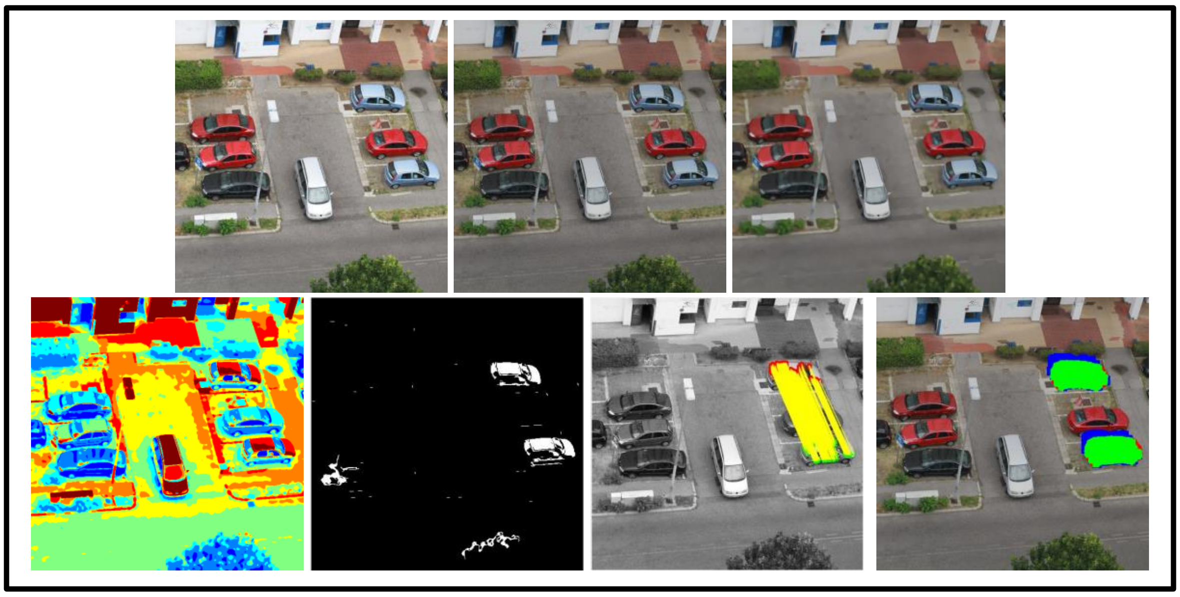

Figure 5-29: (top to bottom, left to right) Input forged image with copied object rotated by 5°, Forged image after Contrast Adjustment, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

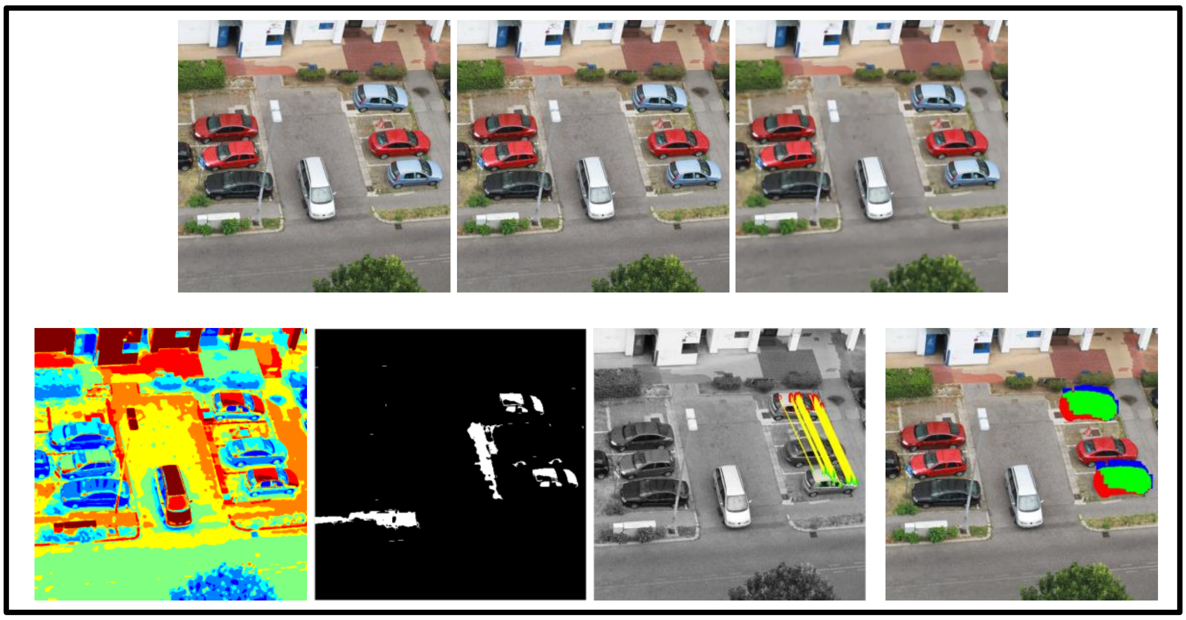

Figure 5-30: (top to bottom, left to right) Input forged image with copied object rotated by 5°, Forged image after Colour Reduction, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

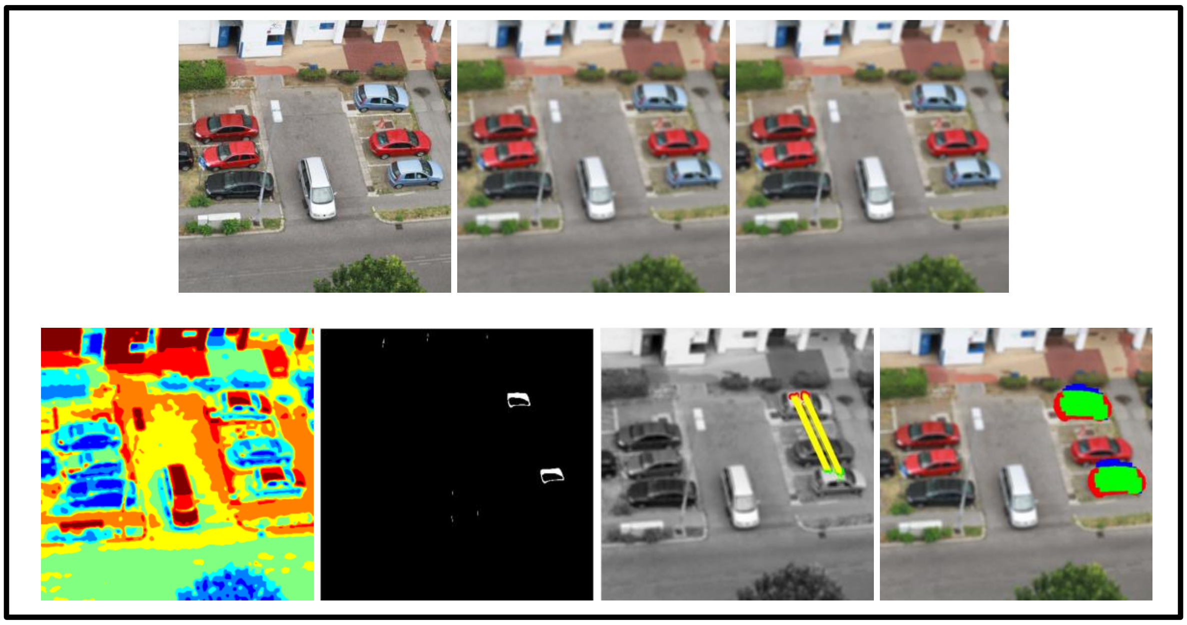

Figure 5-31: (top to bottom, left to right) Input forged image with copied object rotated by 5°, Forged image after Image Blurring, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

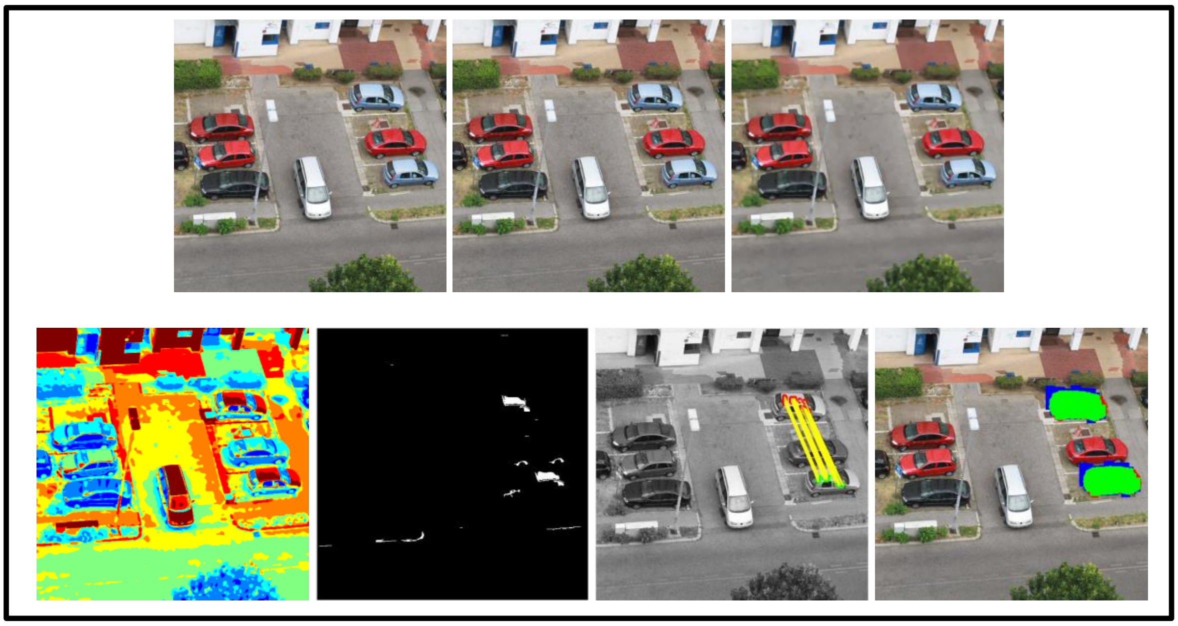

Figure 5-32: (top to bottom, left to right) Input forged image with copied object rotated by 5°, Forged image after Adding Noise, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

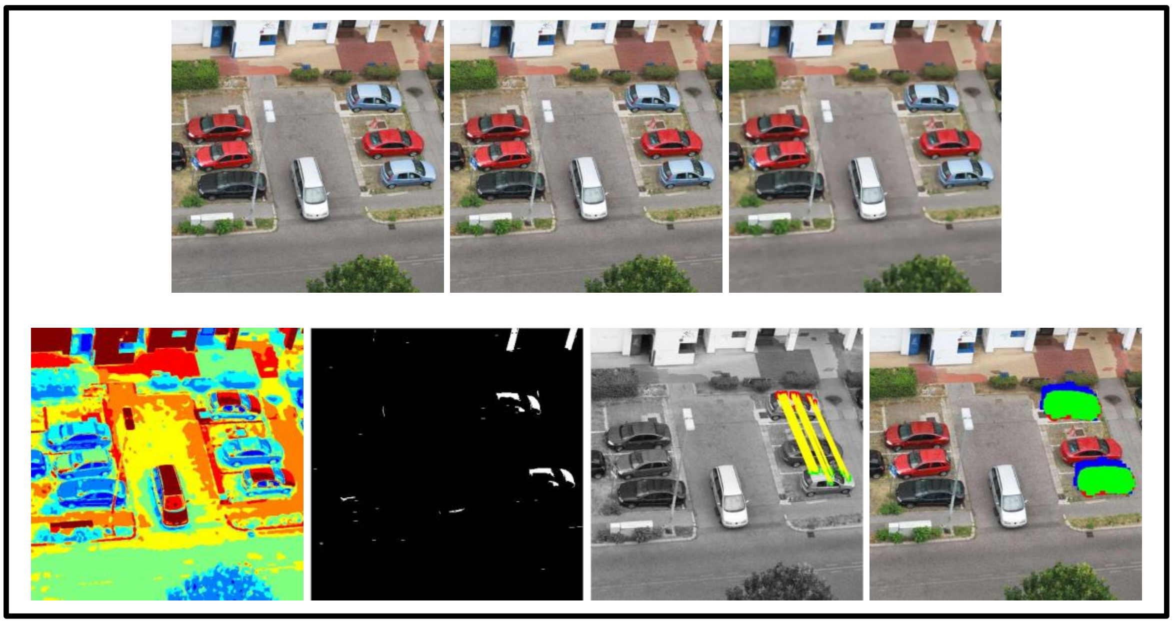

Figure 5-33: (top to bottom, left to right) Input forged image with copied object rotated by 5°, Forged image after JPEG Compression, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

Table 5-2 CMF detection with Rotation and post-processing (Attacks)

Post-processing	**F-measure**	**Number of Detected images out of 40**
Rotation without attack	0.71	35
Brightness Change Range(0.01, 0.8)	0.7	35
Contrast Adjustment Range (0.01,0.8)	0.7	35
Colour Reduction (32 intensity levels)	0.7	35
Image Blurring (5×5 average filter)	0.7	35
Adding Noise (μ=0,σ^2=0.0005)	0.64	31
JPEG Compression (quality factor=40)	0.69	33

5.4.6 Analysis of the proposed algorithm to detect CMF using the Rolling Guidance Filter and Multi-thresholding

It is not possible to use the SIFT/DSIFT to describe the irregular blocks (segments). We used the Segment Gradient Orientation Histogram (SGOH), inspired by SIFT, to describe each segment (irregular block)[102]. The experimental work illustrates that the performance of this suggested algorithm (F-measure = 0.79) is much better than that of our previously proposed algorithms, in detecting plain duplication (translation), which detected on 36 out of 40 image. The under-segmentation is the main reason that the proposed algorithm cannot detect forgery on some images, see Figure 5-34.

The suggested method was designed to detect copy-rotate-move forgery in images, because each segment is rotated to its canonical orientation before computing the Segment Gradient Orientation Histogram (SGOH).

The interpolation adds changes to the Copy-Rotate-Move (CRM) objects (segments) which makes the detection of the CRM forgery much harder on the segment level than the block level. The reason is that the bigger the block, the more interpolation changes can be added, and the more the differences between the copy-rotate-move objects. The algorithm successfully detected CRM forgery on 35 out of 40 images and the F-measure = 0.71.

Still, this method is less complicated and much faster than the previous suggested methods. It takes about 50 seconds to process one image, see Table 5-3.

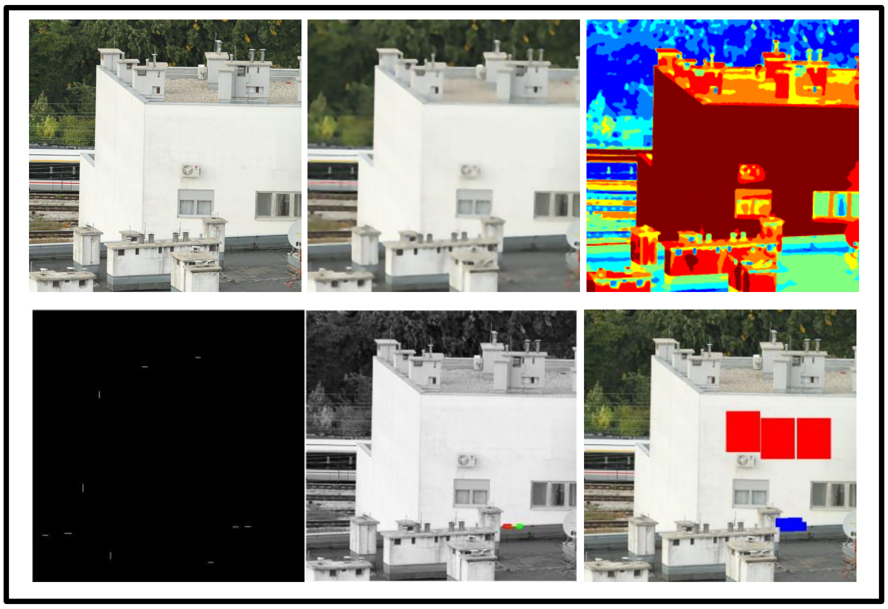

Figure 5-34: (top to bottom, left to right) Input forged image, Rolling Guidance image, Quantized image, Primary detection, RANSAC result and final result after hysteresis.

Table 5-3 Comparison between different segmenation methods

Method	**F-measure**	**Number of Detected images out of 40**	**Run time for one image**
Superpixel (translation)	0.14	21	~ 27 min.
Bag of Words Image (translation)	0.46	25	~ 15 min.
Rolling Guidance (translation)	0.79	36	~ 50 sec.
Rolling Guidance (rotation)	0.71	35	~ 50 sec.

5.5 Conclusions

This chapter has described an approach to describe image segments by dense descriptors. Experimentally, we found when features extracted densely from the segment, the segmentation result has a significant influence on the feature vectors that are produced. Therefore, the main challenge with the segmentation based CMF detection is to segment the copy-move objects consistently.

Three different methods have been used to detect CMF. The SLIC superpixels method and Bag of Words Image method cannot segment the copy-move objects consistently and produced weak results, see sections 5.2.3 and 5.3.3. Moreover, these methods take long time to process single 512×512 image, see Table 5-3. On the other hand, Rolling Guidance filter and multi-thresholding method segments the copy-move objects in a more consistent way than SLIC or BoWImage can do. Combining this proposed method with the SGOH and 3DRGB produces very good results with a short time to process a single image, see Table 5-3.

Chapter 6
Summary, Conclusions & Future Work

6.1 Summary

We have suggested three different methods to detect and localize CMF. The improved DSIFT and FFRID are block-based methods and can detect CMF very well. The SGOH is a segmentation-based method, and it detects CMF forgery very good.

The FFRID requires 65 sec. to process an image of size 512×512 while the improved DSIFT needs 220 sec. to process the same image. The FFRID has low computation cost and produces similar results to improved DSIFT on plain CMF. While the improved DSIFT is more robust to complex transformations (rotation, scaling and combined transformation) than FFRID. The SGOH is faster to compute than FFRID, but it has the under-segmentation problem on flat regions. Overall, we have found that the FFRID works better than other suggested methods, see Table 6-1.

Table 6-1 A comparison between three developed methods in CMF, CMF and post-Processing (Avg. of F-Measure and Number of Detected Images).

Transformation	**Post-processing**	**improved DSIFT**	**FFRID**	**SGOH**	**improved DSIFT NoDI**	**FFRID NoDI**	**SGOH NoDI**
Translation	No-Post processing	**0.93**	**0.93**	0.79	40	40	37
	Image Blurring, (5×5 average filter)	0.78	**0.84**	0.79	38	39	37
	Brightness Change Range (0.01, 0.8)	0.87	**0.92**	0.78	40	40	37
	Colour Reduction (32 intensity levels)	0.90	**0.93**	0.78	40	40	37
	JPEG Compression (quality factor=40)	0.78	**0.90**	0.78	40	40	37
	Contrast Adjustment Range (0.01,0.8)	0.90	**0.93**	0.78	40	40	37
	White Gaussian Noise	0.68	**0.82**	0.69	36	38	32
Rotation	No-Post processing	**0.85**	0.76	0.71	40	40	34
	Image Blurring, (5×5 average filter)	**0.71**	0.67	0.70	36	36	34
	Brightness Change Range (0.01, 0.8)	**0.70**	0.66	**0.70**	36	36	34
	Colour Reduction (32 intensity levels)	**0.70**	0.65	**0.70**	36	35	34
	JPEG Compression (quality factor=40)	0.68	0.63	**0.69**	32	35	34
	Contrast Adjustment Range (0.01,0.8)	0.70	**0.72**	0.70	36	40	34
	White Gaussian Noise	**0.65**	0.61	0.64	31	28	29
Scaling	No-Post processing	**0.40**	0.35	-	28	24	-
Distortion	No-Post processing	0.6	**0.7**	-	36	38	-
Combination	No-Post processing	**0.57**	0.45	-	34	26	-

6.2 Conclusions

In this thesis, three different methods were developed for detecting and localizing copy-move forgery. We also developed a method to estimate the similarity threshold between feature vectors and optimize the generated results using hysteresis thresholding.

In the first CMF method, we improved the DSIFT by using the intensity centroid measure method to estimate patch orientation, instead of the standard SIFT method. This made our improved DSIFT rotation invariant without multiple orientation assignment for the same patch. In the improved DSIFT, we used circular blocks instead of square ones to eliminate the border effect; using circular shaped Gaussian weighting in the standard SIFT can reduce the effects of square blocks, but cannot totally eliminate the edge effects as circular blocks do. Experimentally, we found that the improved DSIFT is more robust to rotation than the Zernike moments (see section 3.3.3), and is highly discriminative. It can detect forgery in flat regions and structurally complex images.

In the second CMF method, we considered the image as a surface, and used the least-squares method to fit the parameters of a quadratic function that represents the surface of each block in the image. The generated 4-element Fitting Function Rotation Invariant Descriptor (FFRID) is rotation-invariant, because we used the intensity centroid measure method to rotate each block to its canonical orientation. The experimental results show that although the FFRID is only a four element feature vector it is discriminative and can detect forgery in flat regions and structurally complex images. The FFRID is more robust to post-processing operations and requires less computational time than the improved DSIFT or Zernike moments. Finally, we found that the feature descriptors with fewer elements speed up the matching to find potential copy-move objects.

In the third CMF method, we tested three different segmentation methods for CMF detection.

We first tried to use the SLIC superpixels in CMF detection. The size of copy-move objects can occupy small or large parts of the image; therefore the required number of approximately similar-sized superpixels (K) differs from one image to another. To overcome this problem, we tested differrent K and found that in different cases we should use different K. The Bag of Visual Words with the LBP and a 3D colour histogram were used to describe each segment. The 2ANN in a K-d tree have been found for each BoWFV to determine the potential CMF, but experimentally we found that the SLIC cannot segment the copy-move objects consistently. The experimental results illustrate the unreliability of this method, due to the significant influence that the segmentation result/error has on the generated feature vectors when the features are extracted densely from each segment.

In the second segmentation method, we developed the Bag of Words Image segmentation as a potential solution for CMF detection. We produced the LBP image by computing the LBP for each pixel in the image and assigned the LBP value to the centre pixel. A 256-histogram feature vector was used to describe each 16×16 overlapping block in the LBP image. We used the K-means++ clustering to find 16 centres in the feature vectors. Similar feature vectors were clustered together, such that the minimum Euclidean distance between each feature vector and the K centres indicated which centre the feature belonged to. Then we used this centre number to label the centre pixel of each block.

The generated Bag of Words Image clustered (segmented) the copy-move objects in an almost similar way. To describe each segment, we used the HOG, the Hu moments and the size of each segment. The 2ANN in a K-d tree were found for each concatenated feature vector. The experimental results illustrate that the performance of this suggested algorithm is better than that of our previous proposed algorithm (SLIC superpixel). But the BoWImage cannot segment the copy-move objects consistently; it produces segmentation errors especially in flat regions and similar textures.

In the third segmentation method, we used the Rolling Guidance filter to smooth the image and the Otsu method to find 7 different thresholds on the filtered image. Then we used these 7 thresholds to quantize the Rolling Guidance filtered image into 8 different labels. We developed the Segment Gradient Orientation Histogram (SGOH) to describe the gradient and used the 3D colour histogram to describe the colour distribution of each segment. Then we built a K-d tree to all the concatenated feature vectors to find 2ANN which showed Euclidean distance of less than a threshold.

The experimental work illustrates the very good performance of this method. The Rolling Guidance filter and multi-thresholding segmentation method can segment the copy-move objects in more consistent way than SLIC segmentation for CMF/CRMF. The SGOH which was inspired by SIFT can describe each segment (irregular block) effectively. But this method still has the limitation of under-segmentation; it cannot detect forgery on flat regions.

We found that the similarity threshold between feature vectors is one of the most important parameters in detecting CMF. Consider that this threshold differs from one image to another. We developed a new method to detect the optimal threshold for each image by optimizing a cost function based on probability distributions of the correct matching of a patch with its rotated and scaled counterpart and the false matching of different patches. The experimental work illustrated the performance of this proposed method. We used the optimal threshold with the hysteresis to decrease the number of false matches and produce the best possible result.

6.3 Future work

In the following, we present some of the ideas we intend to pursue for future works.

1. The improved DSIFT and FFRID used the moment method (intensity centroid measure) to find the canonical orientation of each block. Then each block rotated to the estimated orientation which produced a rotation invariant descriptors. Unfortunately, these descriptors are only robust to moderate scaling and not scale invariant. One possible approach is to use Difference of Gaussians as a scale-space filtering to achieve the scale invariance.
2. This thesis considered one type of image forgery, the copy-move forgery. We will adapt the two developed algorithms in Chapter 3 and Chapter 4 to tackle the splicing by finding the dissimilarity between the extracted features.
3. We will use the developed descriptors (improved DSIFT, FFRID and SHOG) in other applications of object detection and localization, e.g. medical image analysis, face detection, etc.
4. Study the effect of using HSV colour space instead of RGB colour space on CMF with the segmentation method which described in Section 5.4.
5. Deep learning uses neural networks to learn useful representations of features directly from data. Perform supervised learning with series

and Directed Acyclic Graph (DAG) Convolutional Neural Networks (CNNs or ConvNets) for classification and regression. Recent advances in deep learning have improved to the point where deep learning outperforms humans in some tasks like classifying objects in images. We will use deep learning (semantic segmentation) to detect and localize CMF. Semantic segmentation describes the process of associating each pixel of an image with a class label, (such as CMF parts, unforged part).

REFERENCES

[1] A. Piva, “An Overview on Image Forensics,” *ISRN Signal Processing*, Vol. 2013, PP. 1–22, 2013.

[2] M. Tralic, Dijana and Zupancic, Ivan and Grgic, Sonja and Grgic, “CoMoFoD - New Database for Copy-Move Forgery Detection,” in *ELMAR, 55th international symposium*, 2013, PP. 49–54.

[3] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, Vol. 60, No. 2, PP. 91–110, Nov. 2004.

[4] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded Up Robust Features,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 3951 LNCS, 2006, PP. 404–417.

[5] J. Fridrich , D. Soukal, J. Lukáš, “Detection of copy-move forgery in digital images,” in *Digital Forensic Research Workshop*, 2003, Vol. 3, PP. 272–276.

[6] Y. Li, “Image copy-move forgery detection based on polar cosine transform and approximate nearest neighbor searching,” *Forensic Science International*, Vol. 224, No. 1–3, PP. 59–67, Jan. 2013.

[7] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, “SLIC Superpixels Compared to State-of-the-Art Superpixel Methods,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 11, PP. 2274–2282, Nov. 2012.

[8] D. Comaniciu and P. Meer, “Mean shift: a robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 5, PP. 603–619, May 2002.

[9] H. Farid, “Digital doctoring: Can we trust photographs?,” *Deception: From ancient empires to Internet dating*, PP. 95–108, 2009.

[10] W. N. Nathalie Diane, S. Xingming, and F. K. Moise, “A Survey of Partition-Based Techniques for Copy-Move Forgery Detection,” *The Scientific World Journal,* Vol. 2014, PP. 1–13, 2014.

[11] H. Farid, “Image Forgery Detection A survey,” *Ieee Signal Processing Magazine,* Vol. 26, No. 2, PP. 16–25, 2009.

[12] S. Baboo, “Automated Forensic Method for Copy-Move Forgery Detection based on Harris Interest Points and SIFT Descriptors.,” *International Journal of Computer Applications*, Vol. 27, No. 3, PP. 9–17, 2011.

[13] Wei Wang, J. Dong, and T. Tan, “Effective image splicing detection based on image chroma,” in *2009 16th IEEE International Conference on Image Processing (ICIP),* 2009, PP. 1257–1260.

[14] H. Farid, “Creating and Detecting Doctored and Virtual Images: Implications to The Child Pornography Prevention Act,” *Department of Computer Science, Dartmouth College,* Vol. 13, 2004.

[15] V. Savchenko, N. Kojekine, and H. Unno, “A practical image retouching method,” in *First International Symposium on Cyber Worlds, 2002. Proceedings.*, 2002, PP. 480–487.

[16] H. Farid, *Photo Forensics*. MIT, 2016.

[17] T.-T. Ng and S. Chang, “A Data Set of Authentic and Spliced Image Blocks,” 2004.

[18] J. Dong, W. Wang, and T. Tan, “CASIA Image Tampering Detection Evaluation Database,” in *2013 IEEE China Summit and International Conference on Signal and Information Processing*, 2013, PP. 422–426.

[19] D. B. A. and S. G. Amerini I., Ballan L., Caldelli R., “A SIFT-Based Forensic Method for Copy–Move Attack Detection and Transformation Recovery,” *IEEE Transactions on Information Forensics and Security,* Vol. 6, No. 3, PP. 1099 – 1110, 2011.

[20] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, “An

Evaluation of Popular Copy-Move Forgery Detection Approaches," *IEEE Transactions on Information Forensics and Security,* Vol. 7, No. 6, PP. 1841–1854, Dec. 2012.

[21] D. Cozzolino, G. Poggi, and L. Verdoliva, "Efficient Dense-Field Copy-Move Forgery Detection," *IEEE Transactions on Information Forensics and Security,* Vol. 10, No. 11, PP. 2284–2297, Nov. 2015.

[22] B. Wen, Y. Zhu, R. Subramanian, T.-T. Ng, X. Shen, and S. Winkler, "COVERAGE — A novel database for copy-move forgery detection," in *2016 IEEE International Conference on Image Processing (ICIP),* 2016, PP. 161–165.

[23] D. Tralic, I. Zupancic, S. Grgic, and M. Grgic, "CoMoFoD - New Database for Copy-Move Forgery Detection," in *Proceedings of 55th International Symposium ELMAR-2013,* 2013, No. September, PP. 25–27.

[24] D. Powers, "Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation," *Journal of Machine Learning Technologies*, Vol. 2, No. 1, PP. 37–63, 2011.

[25] J. Fridrich, D. Soukal, and J. Lukáš, "Detection of Copy-Move Forgery in Digital Images," *International Journal,* Vol. 3, No. 2, PP. 652–663, 2003.

[26] H. Huang, W. Guo, and Y. Zhang, "Detection of Copy-Move Forgery in Digital Images Using SIFT Algorithm," in *2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application*, 2008, Vol. 2, PP. 272–276.

[27] J. Li, X. Li, B. Yang, and X. Sun, "Segmentation-Based Image Copy-Move Forgery Detection Scheme," *IEEE Transactions on Information Forensics and Security,* Vol. 10, No. 3, PP. 507–518, Mar. 2015.

[28] A. C. Popescu and H. Farid, "Exposing digital forgeries by detecting traces of resampling," *IEEE Transactions on Signal Processing*, Vol.

53, No. 2, PP. 758–767, Feb. 2005.

[29] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, Vol. 2, No. 4, PP. 433–459, Jul. 2010.

[30] Weiqi Luo, Jiwu Huang, and Guoping Qiu, "Robust Detection of Region-Duplication Forgery in Digital Image," in *18th International Conference on Pattern Recognition (ICPR'06)*, 2006, Vol. 4, No. c, PP. 746–749.

[31] H. Lin, C. Wang, and Y. Kao, "Fast Copy-Move Forgery Detection," *Signal Processing*, Vol. 5, No. 5, PP. 188–197, 2009.

[32] M. Zagha and G. E. Blelloch, "Radix sort for vector multiprocessors," in *Proceedings of the 1991 ACM/IEEE conference on Supercomputing - Supercomputing '91*, 1991, PP. 712–721.

[33] B. Xu, G. Liu, and Y. Dai, "A fast image copy-move forgery detection method using phase correlation," *4th International Conference on Multimedia and Security, MINES 2012*, No. 315, PP. 319–322, 2012.

[34] G. Lynch, F. Y. Shih, and H.-Y. M. Liao, "An efficient expanding block algorithm for image copy-move forgery detection," *Information Sciences*, Vol. 239, PP. 253–265, Aug. 2013.

[35] L. Li, S. Li, and J. Wang, "Copy-move forgery detection based on PHT," in *2012 World Congress on Information and Communication Technologies*, 2012, PP. 1061–1065.

[36] L. Li, S. Li, G. Wang, and A. Abraham, "An evaluation on circularly orthogonal moments for image representation," in *International Conference on Information Science and Technology*, 2011, No. 4, PP. 394–397.

[37] Pew-Thian Yap, Xudong Jiang, and A. C. Kot, "Two-Dimensional Polar Harmonic Transforms for Invariant Image Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.

32, No. 7, PP. 1259–1270, Jul. 2010.

[38] H. Shao, T. Yu, M. Xu, and W. Cui, “Image region duplication detection based on circular window expansion and phase correlation,” *Forensic Science International*, Vol. 222, No. 1–3, PP. 71–82, Oct. 2012.

[39] M. Slaney and M. Casey, “Locality-Sensitive Hashing for Finding Nearest Neighbors [Lecture Notes],” *IEEE Signal Processing Magazine*, Vol. 25, No. 2, PP. 128–131, Mar. 2008.

[40] X. Bi, C.-M. Pun, and X.-C. Yuan, “Multi-Level Dense Descriptor and Hierarchical Feature Matching for Copy–Move Forgery Detection,” *Information Sciences*, Vol. 345, PP. 226–242, Jun. 2016.

[41] Y. Wo, K. Yang, G. Han, H. Chen, and W. Wu, “Copy–move forgery detection based on multi-radius PCET,” *IET Image Processing*, Vol. 11, No. 2, PP. 99–108, Feb. 2017.

[42] V. Christlein, C. Riess, and E. Angelopoulou, “On rotation invariance in copy-move forgery detection,” in *2010 IEEE International Workshop on Information Forensics and Security*, 2010, PP. 1–6.

[43] S. Ketenci and G. Ulutas, “Copy-move forgery detection in images via 2D-Fourier Transform,” in *2013 36th International Conference on Telecommunications and Signal Processing (TSP)*, 2013, PP. 813–816.

[44] S.-J. Ryu, M.-J. Lee, and H.-K. Lee, “Detection of Copy-Rotate-Move Forgery Using Zernike Moments,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 6387 LNCS, 2010, PP. 51–65.

[45] S. J. Ryu, M. Kirchner, M. J. Lee, and H. K. Lee, “Rotation invariant localization of duplicated image regions based on zernike moments,” *IEEE Transactions on Information Forensics and Security*, Vol. 8, No. 8, PP. 1355–1370, Aug. 2013.

[46] Le Zhong and Weihong Xu, “A robust image copy-move forgery detection based on mixed moments,” in *2013 IEEE 4th International Conference on Software Engineering and Service Science*, 2013, No. 208098, PP. 381–384.

[47] Y. Hsu, H. Arsenault, and G. April, “Rotation-invariant digital pattern recognition using circular harmonic expansion,” *Applied Optics*, 1982.

[48] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, “PatchMatch,” *ACM Transactions on Graphics*, Vol. 28, No. 3, P. 1, Jul. 2009.

[49] M. R. Teague, “Image analysis via the general theory of moments*,” *Journal of the Optical Society of America*, Vol. 70, No. 8, P. 920, Aug. 1980.

[50] Y. Sheng and H. Arsenault, “Experiments on pattern recognition using invariant Fourier–Mellin descriptors,” *JOSA A*, 1986.

[51] J.-C. Lee, “Copy-move image forgery detection based on Gabor magnitude,” *Journal of Visual Communication and Image Representation*, Vol. 31, PP. 320–334, Aug. 2015.

[52] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 7, PP. 971–987, Jul. 2002.

[53] M. Hussain, G. Muhammad, S. Q. Saleh, A. M. Mirza, and G. Bebis, “Copy-Move Image Forgery Detection Using Multi-Resolution Weber Descriptors,” in *2012 Eighth International Conference on Signal Image Technology and Internet Based Systems*, 2012, PP. 395–401.

[54] M. Akbarpour Sekeh, M. A. Maarof, M. F. Rohani, and B. Mahdian, “Efficient image duplicated region detection model using sequential block clustering,” *Digital Investigation*, Vol. 10, No. 1, PP. 73–84, Jun. 2013.

[55] V. Christlein, C. Riess, and E. Angelopoulou, “A Study on Features for the Detection of Copy-Move Forgeries,” in *Sicherheit 2010, Gesellschaft für Informatik e. V.*, 2010, PP. 105–116.

[56] E. Ardizzone, A. Bruno, and G. Mazzola, “Detecting multiple copies in tampered images,” in *2010 IEEE International Conference on Image Processing*, 2010, PP. 2117–2120.

[57] X. Pan and S. Lyu, “Region duplication detection using image feature matching,” in *IEEE Transactions on Information Forensics and Security*, 2010, Vol. 5, No. 4, PP. 857–867.

[58] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, Vol. 24, No. 6, PP. 381–395, Jun. 1981.

[59] T. Ng, S. Chang, J. Hsu, and M. Pepeljugoski, “Columbia Photographic Images and Photorealistic Computer Graphics Dataset,” *Columbia University, ADVENT Tecnical Report # 205-2004-5*, PP. 1–23, 2005.

[60] L. Chen, W. Lu, J. Ni, W. Sun, and J. Huang, “Region duplication detection based on Harris corner points and step sector statistics,” *Journal of Visual Communication and Image Representation*, Vol. 24, No. 3, PP. 244–254, Apr. 2013.

[61] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” in *Procedings of the Alvey Vision Conference 1988*, 1988, PP. 23.1–23.6.

[62] J. S. Beis and D. G. Lowe, “Shape indexing using approximate nearest-neighbour search in high-dimensional spaces,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, PP. 1000–1006.

[63] “True Color Kodak Images.” [Online]. Available:

http://r0k.us/graphics/kodak/. [Accessed: 03-Mar-2017].

[64] J.-M. Guo, Y.-F. Liu, and Z.-J. Wu, "Duplication forgery detection using improved DAISY descriptor," *Expert Systems with Applications*, Vol. 40, No. 2, PP. 707–714, Feb. 2013.

[65] M. Brown, R. Szeliski, and S. Winder, "Multi-Image Matching Using Multi-Scale Oriented Patches," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2008, Vol. 1, PP. 510–517.

[66] E. Tola, V. Lepetit, and P. Fua, "DAISY: An efficient dense descriptor applied to wide-baseline stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, No. 5, PP. 815–830, May 2010.

[67] G. Schaefer and M. Stich, "UCID - An Uncompressed Colour Image Database," in *Proc. SPIE 5307, Storage and Retrieval Methods and Applications for Multimedia*, 2003, PP. 472–480.

[68] M. Jaberi, G. Bebis, M. Hussain, and G. Muhammad, "Accurate and robust localization of duplicated region in copy–move image forgery," *Machine Vision and Applications*, Vol. 25, No. 2, PP. 451–475, Feb. 2014.

[69] X. Guo, X. Cao, J. Zhang, and X. Li, "MIFT: A Mirror Reflection Invariant Feature Descriptor," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 5995 LNCS, No. PART 2, 2010, PP. 536–545.

[70] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, L. Del Tongo, and G. Serra, "Copy-move forgery detection and localization by means of robust clustering with J-Linkage," *Signal Processing: Image Communication*, Vol. 28, No. 6, PP. 659–669, Jul. 2013.

[71] R. Toldo and A. Fusiello, "Robust multiple structures estimation with

J-linkage," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 5302 LNCS, No. PART 1, PP. 537–547, 2008.

[72] A. Hartley, R.~I. and Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[73] E. Silva, T. Carvalho, A. Ferreira, and A. Rocha, "Going deeper into copy-move forgery detection: Exploring image telltales via multi-scale analysis and voting processes," *Journal of Visual Communication and Image Representation*, Vol. 29, PP. 16–32, May 2015.

[74] A. Haar, "Zur Theorie der orthogonalen Funktionensysteme," *Mathematische Annalen*, Vol. 71, No. 1, PP. 38–53, Mar. 1911.

[75] E. Ardizzone, A. Bruno, and G. Mazzola, "Copy – Move Forgery Detection by Matching Triangles of Keypoints," *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 10, PP. 2084–2094, Oct. 2015.

[76] R. Dyer, H. Zhang, and M. Torsten, "A survey of Delaunay structures for surface representation," PP. 1–47, 2009.

[77] Y. Zhu, X. Shen, and H. Chen, "Copy-move forgery detection based on scaled ORB," *Multimedia Tools and Applications*, Vol. 75, No. 6, PP. 3221–3233, Mar. 2016.

[78] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, 2005, Vol. II, PP. 1508–1515 Vol. 2.

[79] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, 2011, PP. 2564–2571.

[80] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, "An evaluation of popular copy-move forgery detection approaches," *IEEE*

Transactions on Information Forensics and Security, Vol. 7, No. 6, PP. 1841–1854, Dec. 2012.

[81] C.-M. Pun, X.-C. Yuan, and X.-L. Bi, "Image Forgery Detection Using Adaptive Oversegmentation and Feature Point Matching," *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 8, PP. 1705–1716, Aug. 2015.

[82] Bhavya Bhanu M P and Arun Kumar M N, "Copy-move forgery detection using segmentation," in *2017 11th International Conference on Intelligent Systems and Control (ISCO)*, 2017, PP. 224–228.

[83] R. Sekhar and R. S. Shaji, "A Study on Segmentation-Based Copy-Move Forgery Detection Using DAISY Descriptor," in *Advances in Intelligent Systems and Computing*, Vol. 397, L. P. Suresh and B. K. Panigrahi, Eds. New Delhi: Springer India, 2016, PP. 223–233.

[84] I. Rey-otero and M. Delbracio, "Anatomy of the SIFT Method," *Image Processing On Line*, Vol. 6, PP. 8–26, 2014.

[85] A. Bosch, A. Zisserman, and X. Munoz, "Image Classification using Random Forests and Ferns," in *2007 IEEE 11th International Conference on Computer Vision*, 2007, PP. 1–8.

[86] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005, Vol. 1, PP. 886–893.

[87] S.-K. Hwang and W.-Y. Kim, "A novel approach to the fast computation of Zernike moments," *Pattern Recognition*, Vol. 39, No. 11, PP. 2065–2076, Nov. 2006.

[88] A. Khotanzad and Y. H. Hong, "Invariant image recognition by Zernike moments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 5, PP. 489–497, May 1990.

[89] Yong-Sung Kim and Whoi-Yul Kim, "Content-based trademark

retrieval system using visually salient features," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 16, PP. 307–312, 1997.

[90] Lizhi Wang and G. Healey, "Using Zernike moments for the illumination and geometry invariant classification of multispectral texture," *IEEE Transactions on Image Processing*, Vol. 7, No. 2, PP. 196–203, 1998.

[91] Whoi-Yul Kim and Young-Sung Kim, "Robust rotation angle estimator," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 8, PP. 768–773, 1999.

[92] R. J. Prokop and A. P. Reeves, "A survey of moment-based techniques for unoccluded object representation and recognition," in *CVGIP: Graphical Models and Image Processing*, 1992, Vol. 54, No. 5, PP. 438–460.

[93] P. L. Rosin, "Measuring Corner Properties," *Computer Vision and Image Understanding*, Vol. 73, No. 2, PP. 291–307, Feb. 1999.

[94] Z. Xu, Y. Liu, S. Du, P. Wu, and J. Li, "DFOB: Detecting and describing features by octagon filter bank for fast image matching," *Signal Processing: Image Communication*, Vol. 41, PP. 61–71, Feb. 2016.

[95] D. W. Scott, "Scott's rule," *WIREs Comp Stat*, Vol. 2, No. 4, PP. 497–502, Jul. 2010.

[96] M. Muja and D. G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," in *International Conference on Computer Vision Theory and Applications (VISAPP '09)*, 2009, PP. 331–340.

[97] M. Zuliani, "RANSAC for Dummies," *Vision Research Lab, University of California, Santa Barbara*, Vol. 25, No. 2, PP. 164–176, 2008.

[98] V. Christlein, C. Riess, and J. Jordan, “An evaluation of popular copy-move forgery detection approaches,” *... and Security, IEEE ...*, 2012.

[99] A. R. H. Khayeat, X. Sun, and P. L. Rosin, “Improved DSIFT Descriptor Based Copy-Rotate-Move Forgery Detection,” in *Image and Video Technology*, Vol. 8334, No. November, F. Huang and A. Sugimoto, Eds. Berlin, Heidelberg: Springer International Publishing, 2016, PP. 642–655.

[100] J. Canny, “A Computational Approach to Edge Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 6, PP. 679–698, Nov. 1986.

[101] T. Zhang, T. E. Boult, and R. C. Johnson, “Two thresholds are better than one,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, PP. 1–8.

[102] A. R. H. Khayeat, P. L. Rosin, and X. Sun, “Copy-Move Forgery Detection Using the Segment Gradient Orientation Histogram,” Vol. 10270, P. Sharma and F. M. Bianchi, Eds. Cham: Springer International Publishing, 2017, PP. 209–220.

[103] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999, Vol. 2, No. [8, PP. 1150–1157 vol.2.

[104] S. Zhong, Y. Liu, and Q. Chen, “Visual orientation inhomogeneity based scale-invariant feature transform,” *Expert Systems with Applications*, Vol. 42, No. 13, PP. 5658–5667, Aug. 2015.

[105] M. Brown and D. G. Lowe, “Automatic Panoramic Image Stitching using Invariant Features,” *International Journal of Computer Vision*, Vol. 74, No. 1, PP. 59–73, Apr. 2007.

[106] I. Skrypnyk and D. G. Lowe, “Scene Modelling, Recognition and Tracking with Invariant Image Features,” in *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2004, No.

Ismar, PP. 110–119.

[107] Xinghua Sun, Mingyu Chen, and A. Hauptmann, “Action recognition via local descriptors and holistic features,” in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2009, PP. 58–65.

[108] G. Z. - and H. W. -, “SURF-based Detection of Copy-Move Forgery in Flat Region,” *International Journal of Advancements in Computing Technology*, Vol. 4, No. 17, PP. 521–529, Sep. 2012.

[109] N. Litke, M. Droske, M. Rumpf, and P. Schröder, “An Image Processing Approach to Surface Matching,” *Third Eurographics Symposium on Geometry Processing*, PP. 207–216, 2005.

[110] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision*. 1999.

[111] D. Kleinbaum, L. Kupper, and K. Muller, *Applied regression analysis and other multivariable methods. PWS-Kent, Boston*. 1988.

[112] I. Anjam and J. Valdman, “Fast MATLAB assembly of FEM matrices in 2D and 3D: Edge elements,” *Applied Mathematics and Computation*, Vol. 267, No. 13, PP. 252–263, Sep. 2015.

[113] M. Pietikäinen, A. Hadid, G. Zhao, and T. Ahonen, *Computer Vision Using Local Binary Patterns*, Vol. 40, No. 11. London: Springer London, 2011.

[114] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo, “Evaluating bag-of-visual-words representations in scene classification,” in *Proceedings of the international workshop on Workshop on multimedia information retrieval - MIR ’07*, 2007, Vol. 63, P. 197.

[115] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Communications of the ACM*, Vol. 18, No. 9, PP. 509–517, Sep. 1975.

[116] Ming-Kuei Hu, “Visual pattern recognition by moment invariants,”

IEEE Transactions on Information Theory, Vol. 8, No. 2, PP. 179–187, Feb. 1962.

[117] D. Arthur, D. Arthur, S. Vassilvitskii, and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007, Vol. 8, PP. 1027–1035.

[118] L. G. Haralick, Robert M. and Shapiro, *Computer and Robot Vision*, 1st ed. Addison-Wesley Longman Publishing, 1992.

[119] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 9, No. 1, PP. 62–66, 1979.

[120] P. S. Liao, T. S. Chen, and P. C. Chung, "A fast algorithm for multilevel thresholding," *Journal of Information Science and Engineering*, Vol. 17, No. 5, PP. 713–727, 2001.

[121] Q. Zhang, X. Shen, L. Xu, and J. Jia, "Rolling Guidance Filter," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 8691 LNCS, No. PART 3, 2014, PP. 815–830.

[122] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images," in *International Conference on Computer Vision*, 1998, PP. 839–846.

[123] K. He, J. Sun, and X. Tang, "Guided Image Filtering," in *Link.Springer.Com*, Vol. 6311, No. Chapter 1, 2010, PP. 1–14.