Image Forgery Detection Algorithm Using Particle Swarm Optimization

Hussain Alibrahim
Department of Computer Science
North Dakota State University
Fargo, USA
hussain.alibrahim@ndsu.edu

Simone A. Ludwig
Department of Computer Science
North Dakota State University
Fargo, USA
simone.ludwig@ndsu.edu

***Abstract*—Copy-move forgery is one of the most used manipulations for tampering with digital images. The authenticity of the image becomes more crucial when the images are used in important processes. Key points-based algorithms have been reported to be very effective in revealing copy-move evidence due to their robustness against various attacks. However, these approaches sometimes fail to make good prediction because of different factors such small number of key points detected, or wrongly detected key points. Matching the correct key points and filtering the wrong key points are other difficult tasks. One reason behind these issues is the parameters used to configure the key point detection algorithm. In this paper, another CMF (copy-move forgery) detection algorithm is proposed, by applying particle swarm optimization to find the best parameters for the algorithm for all different phases. Furthermore, filtering is achieved through two stages to remove most of the wrong key points detected. Additionally, triangulation is used as another technique applied to the algorithm in order to increase the detection area. Experimental results shows that the algorithm has good performance.**

***Index Terms*—Image Forgery Detection, CMFD, PSO, DBSCAN, SIFIT.**

I. Introduction

Image modification, editing and tampering have become an easy task with a lot of easy to use and professional software developed for image processing. Forgery images are used widely in social media, but also appear frequently in public media and in daily life. The adverse effects of these forgery images have raised lots of concerns.

Different methods exist to alter images such as image splicing and copy move forgery. The splicing involves more than one image, by copying some object from one image and adding it to the second image. The copy move forgery is applied by using one image only by copying some part of image and pasting it to another location in the image. This process of copy and paste include different type of operations, such as geometric operations that include scaling, rotating, reflecting, translating and affine transformation. The post processing operation is another type of operation applied while image editing that include JPEG compression, adding Gaussian noise, color reduction, contract adjustment, brightness change, and image blurring.

Detect copy move forgery region is the interest in this paper. Lots of copy-move forgery detection (CMFD) solutions were proposed, which can be categorized into three approaches: key point based, block-based, and deep-learning-based [1]. The extraction of picture features, feature matching, filtering of false matching, and some additional processing to disclose the assaults are stages shared by the key point based and block-based methods. The block-based algorithms break up images into overlapping blocks and extract block features by frequently employing invariant moment approaches. The high entropy regions are searched by the key point based algorithms, which then extract the extreme values of each pixel for the entire image. The deep-learning-based algorithms, in contrast, are very new. They use many forged photos to train their neural networks for CMFD during the calculation process. Unlike the other two methods, deep neural networks' workflow does not involve sequential computation phases.

Each of the former detection techniques has some limitations. First, copy-move forgery images cannot be accurately detected by block-based techniques. Second, these techniques do not perform well for detecting the copy-move regions with various geometric and post-processing attack actions. Third, there are a lot of false-positive pixels in the detection results produced by these algorithms. To address these issues, we provide a new CMFD method in this study.

The main contributions are as follows:

1) Filter steps reduce the number of false positive key points that were detected in the first steps.
2) Optimizing the parameters in different phases make the algorithm result better than when using the default values.
3) Thus, the algorithm performance is better than the other CMFD approaches.

The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 presents the copy-move forgery detection algorithm; Section 4 presents the experiments; Section 5 ends with the conclusions.

II. Related Work

The CMFD algorithms can be categorized into three approaches: key point based, block-based, and deep-learning-based. They have different advantages and disadvantages.

1) Key point based algorithms: These algorithms are usually fast and commonly perform well against geometric

attack operations. This performance comes from their ability to find key points regardless if images have been tampered with by different operations such as rotating, translating and affine transformation. This ability of extracting key points provide a solid base for later stages in these algorithms. Different kind of key points techniques exist and are widely used, since the increase of forgery detection algorithm developed. Examples are: FAST (features from accelerated segment test) [2], BRISK (binary robust invariant scalable key points) [3], ORB (oriented BRIEF) [4], SURF (speeded up robust features) [5], SIFT (the scale invariant feature transform key point) [6], and KAZE [7]. However, the strong point of these algorithms has the same weaknesses since a lot of key points detected are false match pairs i.e., invalid key points. Some researchers have suggested algorithms for filtering out the false matching key point pairs to get around this flaw. There are two different categories for these filtering algorithms:

- Color Features Based [8], [9] This type computes the adjacent areas of the key point pairs' color features to decide whether these areas belong to the copy-move regions or not. Due to the lack of geometric invariance of the extracted color features, this type is relatively weak against the geometric attack operations. But robust against the post-processing operations.
- Affine Transformation Based [10] This type computes the rotation degree and scale factor of the whole image and each key point pair; the key point pairs with the same rotation degree and scale factor as the image are considered as the true matching key point pairs. In contrast to color feature based approaches, this type is strong against geometric operations, but not robust against post-processing operations.

2) Block based algorithms: The main idea of these kinds of algorithms is to divide the image into overlapping blocks at the first stage, then to extract the features of these blocks. [11] was one of the early algorithms developed using divided block to detect copy move forgery, which was based on quantized discrete cosine transform (QDCT) technique to extract features. Later many algorithms using different feature extraction methods were proposed such as discrete cosine transform (DCT) [12], blur moment invariants, and undecimated dyadic wavelet transform (DyWt) [13]. The major drawback of these algorithms is their inability to handle geometrical attack operations. To address this weakness, some methods based on invariant moment techniques are proposed, such as Zernike moments [17], polar cosine transform (PCT) [14], polar complex exponential transform (PCET) [15], discrete analytical Fourier-Mellin transform (DAFMT) [16]. These algorithms are generally resistant to its post-processing activities since the recovered features indicate certain important block characteristics based on all the block's pixels. However, the major drawback of these algorithms is the huge computation power needed to divide the image to black and then extract each block's important features which is a complex operation. Furthermore, geometric attacks, such as huge scaling operations that increase copied region size by 100% or 200%, are not adequately addressed.
3) Deep learning based algorithms: Deep learning algorithms have been applied to many problems in the last few years, CMFD is one of these areas. Many examples are available for these algorithm such as BusterNet [18], which proposed two deep learning approaches and an end-to-end deep neural network to detect forgery areas. Another example is AR-Net [19], which is exploiting an image's self-correlation features. Other algorithms used deep convolution neural networks and semantic segmentation to detect the copy-move and splicing forgery images. However, all these solutions' performance depend on the training phase, where data has to be prepared for each module. The major issues with deep learning based algorithms as per [1], is the volume of useful, high-quality data since there could be a lot of copied and pasted objects with unpredictable properties that are not present in the training data. Another issue is, these algorithms frequently have the technical restriction of requiring all input photos to have a particular size. When an image needs to be scaled up or down to fit a new size, a lot of visual information is lost, or noise is introduced. Furthermore, the current algorithms are not performing as block based or key points detection based approaches.

III. Proposed Algorithm

The proposed algorithm is composed of different components which are: detection, matching, filtering, triangulation to identify the copy and paste region. The optimization process is used to find the best value of different parameter in each step. The details of this process is as follows:

A. Detection

Image key points detection methods are used to find the spatial locations, or points in the image that define what is interesting or what stands out in the image. These points can be corners or edges in the image content, blobs or any other image part that describes the image content. Different algorithm exist to detect the key points in the image such as Scale Invariant Feature Transform (SIFT), which is the one used in this paper. SIFIT [20] is fast since the cost of extracting features or key points is minimized by taking a cascade filtering approach, in which the more expensive operations are applied only at locations that pass an initial test. The major stages of computation used to generate the set of image features are:

1) Scale-space extrema detection: All scales and image locations are searched in the initial stage of computation. It is effectively implemented by locating possible interest

locations that are independent of scale and orientation using a difference-of-Gaussian function.

2) Key point localization: A thorough model is fitted to each possible location to establish location and scale. The selection of key points is based on indicators of their stability.
3) Orientation assignment: Based on the local image gradient directions, one or more orientations are assigned to each key point location. The transformations are invariant because all subsequent operations are carried out on picture data that has been altered in relation to the given orientation, scale, and location for each feature.
4) Key point descriptor: The local image gradients are measured at the selected scale in the region around each key point. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

In this phase, four parameters were part of optimization process as listed with simple description:

- Number of Octave Layers: The number of layers in each octave.
- Contrast Threshold: used to filter out weak features in semi-uniform (low-contrast) regions. The larger the threshold, the less features are produced by the detector.
- Edge Threshold: The threshold is used to filter out edge-like features. Note that its meaning is different from the Contrast Threshold, i.e., the larger the edge threshold, the less features are filtered out (more features are retained).
- Sigma: The sigma of the Gaussian applied to the input image at the octave.

B. Matching

The second step is finding the matched key points. Since this is a Copy Move Forgery Detection, this means that some part of the image is copied and moved to another location in same image. As a result, the copied region has the same original region key points. Descriptor of 128 bit length is represented as a feature vector to form the key point descriptor. These vector values measure the key point similarity, so the difference between exact key point is zero. However, descriptor values change if any geometric or post processing operations is applied to the region. The Euclidean distance equation 1 is used to measure the difference between the key points.

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{1}$$

In this step, to find a similar feature vector for feature vector $f1$, first another feature vector $f2$ has to be found that has the smallest distance $L1$, i.e., the smallest distance available, then another feature vector $f3$ has to be identified that is not $f1$ or $f2$ with a second smallest distance $l2$. Finally, if $L1/L2 <$ threshold, then $f1$ and $f2$ are similar key points. Figure 2 shows the key points after matching. The threshold value is one of the optimized parameters in this paper.

C. Filtering

After the matching steps, non-matched key points are removed, however there are always some issues with matching because it depends on the threshold value. So filtering wrong matched key points step is required. The filtering is applied in 2 steps:

1) Grid Based Filter [1]:
   a) For any two blocks i and j, $i \neq j$, let $P_{i,j}$ be the set of key point pairs connecting block i and j, that is: $|P_{i,j}| = k_x, k_y | k_x$is a key point in block i,and k_y is a key point in block j.
   b) Let $|Pi,j|$ be the total number of elements (key point pairs) in $P_{i,j}$.
   c) Given a block j, we call the 9 $(3/time3)$ neighboring blocks, with three rows and three columns, where block j is the center one, as the grid of j. In such a grid, block j is labeled as j^0, while the other 8 surrounding blocks are labeled as $j^1, ..., j^8$, according to a specific order. For example, Figure 1 shows two grids.
   d) For two blocks i and $j, i \neq j$, let $C_{i,j}$ be the number of key point pairs connecting block i and j, plus the number of key point pairs connecting any surrounding neighbor of i and any surrounding neighbor of j given by Equation 2:

$$C_{i,j} = |P_{i^0,j^0}| + \sum_{u=1}^{8}\sum_{v=1}^{8}|P_{i^u,j^v}| \tag{2}$$

   e) For a block j, let nkj be the number of key points in block j.
   f) For two blocks i and $j, i \neq j$, as defined in Equation 3.

$$related(i,j) = \begin{cases} True, & \text{if } C_{i,j} > min(\sqrt[3]{nk_i}, \sqrt[3]{nk_j}) \\ False & \text{otherwise} \end{cases} \tag{3}$$

   In this filtering step, the block size is part of the optimization process.

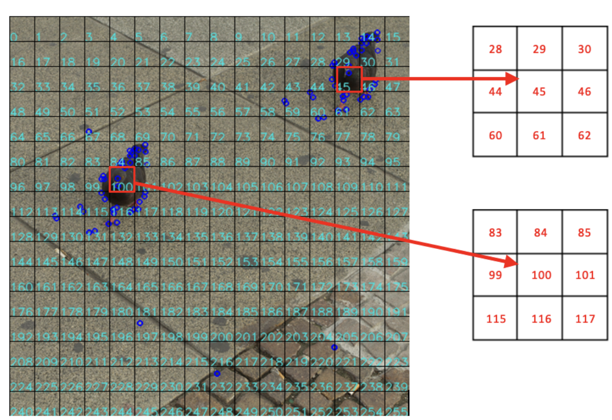


Fig. 1: Grids of two related blocks

2) Density clustering using DBSCSN: After the first part filter step, there are some key points that still exist and are not part of matched group. For this density-based spatial clustering of applications with noise (DBSCAN) it finds core samples of high density and expands clusters from them. It has two main parameters:
   - Eps: The distance that specifies the neighborhoods. Two points are considered to be neighbors if the distance between them are less than or equal to eps.
   - MinPts: Minimum number of data points to define a cluster.

   Based on these parameters' points are classified as core points; Border point or Outlier one. Both of these parameters are part of the optimization process.

D. Triangulation

After the filtering step, the remaining points are the correct matched key points remaining, however, these key points are only points with different sizes. To get the maximum region these key points cover, triangulation is used to connect these points without overlapping. A triangulation of points in set P in the plane is a maximal planar subdivision, which has P as its vertex set.

E. Optimization Process

In each steps in this algorithm, there are some parameter values that affect the accuracy. The list and values ranges for these parameters have been chosen to be tuned before the final experiment with the results listed in Table I is run.

Particle Swarm Optimization (PSO) is used as the optimization algorithm. It was proposed by Kennedy and Eberhart in 1995, to model the social behavior of bird flocking. This is a good algorithm that solves minimization and maximization problems. The estimation parameter values have been tuned as represented in Equation 4.

$$D_{Result} = f(X), X = (x_1, x_2, x_3, x_4, x_5, ...) \tag{4}$$

where X is the input parameter, and $f(x)$ is the proposed algorithm and $Dresult$ is the result of the algorithm. Changing values of X lead to changes in the result, which leads to better results based on the optimization function. As shown in Figure 3, the process starts with the random initialization values between the specified ranges as provided in Table I where each value is listed for each parameter. The swarm size was 50. The first parameter set will be used to do the detection, matching, filtering, and triangulation. After that, a new set of parameters is generated and the algorithm is run for N time steps. At the end, the algorithm will return the best parameter set. In this paper, the N value was 100.

One major part of any optimization process is the evaluation function. In most of the detection algorithms there are three main objectives:

- Maximize number of True matched points (TMK).
- Minimize number of False matched points (FMK).
- Minimize number of missing matched points (Miss-MK).

Therefore, these 3 objectives should be considered while creating the evaluation function. In this paper, Equation 5 from [21] was used as the evaluation function.

$$Pmatch = \frac{TMKt}{TMKt + \phi}$$
$$\phi = \begin{cases} MMKt, & \text{if} \quad MMKt > 10 \\ 10 & \text{if} \quad MMKt \leq 10 \end{cases} \tag{5}$$

where $TMKt$ is the true matched point after the filtering process when compared to the image mask, $MMKt$ is any other pairs of key points, but not including any removed pairs from the filtering step. The mismatching coefficient is used to make the detection process more reliable, in short assuming 2 detection results, the first one has 10 $TMKt$ and 0 $MMKt$, while the second one is 100 $TMKt$ and 1 $MMKt$. So, the $Pmatch$ without the mismatching coefficient for the first one is 100%, the second one is 99%, which means the first detection process is better. However, the opposite is the correct. Therefore, the matching process mismatching coefficient is introduced by applying it the new $Pmatch$ values become 50% for the first detection process and 90% for the second process.

IV. Experiments

A. Dataset

CoMoFoD [22] dataset is Image Database for Copy-Move Forgery Detection with 260 images, 200 in the small image category (512x512), and 60 images in the large image category (3000×2000). Each image has one of these transformations:

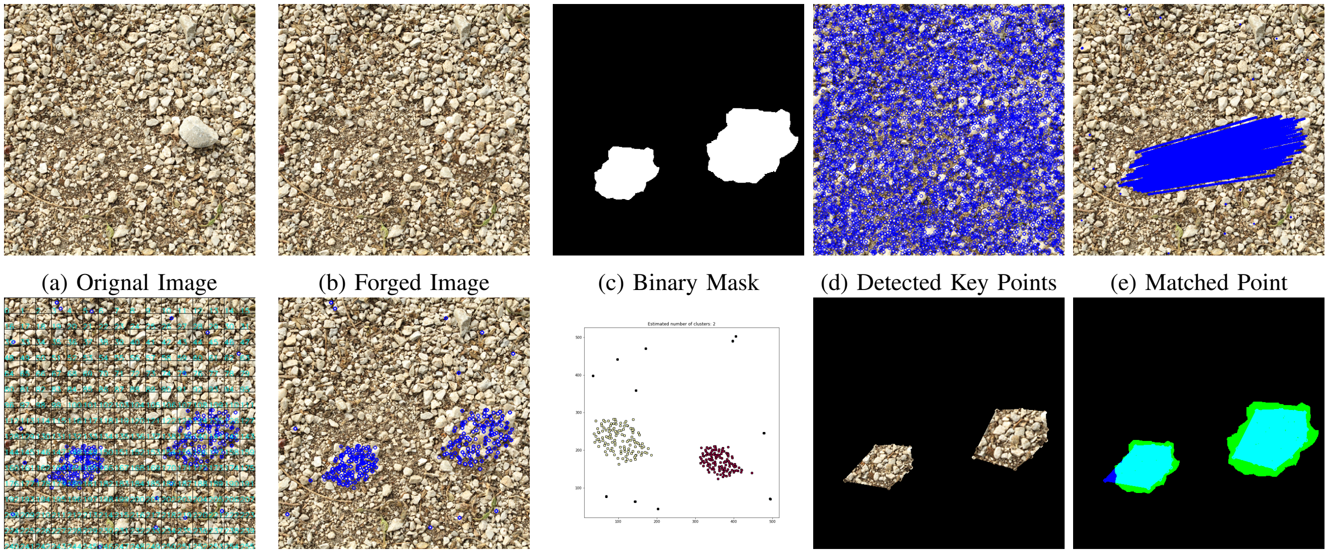
1) Translation - a copied region is only translated to the new location without performing any transformation,
2) Rotation - a copied region is rotated and translated to the new location.
3) Scaling - a copied region is scaled and translated to the new location.
4) Distortion - a copied region is distorted and translated to the new location.
5) Combination - two or more transformations are applied on a copied region before moving it to the new location.

In addition to that, the post processing methods and changes are applied on all original and forged images. The list and description of these methods is listed in Table II:

B. Performance Measures

To evaluate the performance of the algorithm before and after optimization, and compare it with other algorithms, the size of the copy and move region is measured as pixel level. Precision, recall and F1 are the evaluation criteria used in this paper. The equation to calculate these values are given in Equations 6, 7, and 8.

$$Precision = \frac{T_p}{T_p + F_p} \tag{6}$$



(a) Orignal Image (b) Forged Image (c) Binary Mask (d) Detected Key Points (e) Matched Point

(f) Divided Image (g) Image After Filter 1 (h) KP After Filter 2 (i) After Triangulation (j) Comparison

Fig. 2: Steps of Copy Move Forgery Detection Algorithm

TABLE I: Optimized Parameter

Parameter	Phase	Default Value	Search Range	Best Value
nOctaveLayers	Detection	3	[3, 6]	4
contrastThreshold	Detection	0.04	[0.0001, 0.1]	0.063
edgeThreshold	Detection	10	[10, 50]	20
Sigma	Detection	1.6	[1.00, 2.00]	1.3
Matching threshold	Matching	N/A	[0.3, 0.7]	0.61
Block Size	Filtering 1	N/A	8,16,32,64,128	32
Eps	Filtering 2	0.5	[0.5, 60]	37
Min_samples	Filtering 2	5	[5, 80]	61

$$Recall = \frac{T_p}{T_p + F_N} \tag{7}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{8}$$

where TP represents the number of correctly detected pixels; FP represents the number of incorrectly detected pixels; FN represents the number of forged pixels (the pixels in the copy-move regions are all forged) that are not correctly detected.

C. Experiments & Results

The test procedure for this algorithm consists of four steps as follows:

1) The first step in the experiment was to apply the algorithm to all small images category, then take the 50 images with the best result in terms of the F1 value and use them as test data in all other experiments.
2) The next step is the optimization process that is applied to all small images category to find the best parameter for this algorithm.
3) Then, the algorithm has been applied to find the forgery area in the 50 images from Step 1 with parameters achieved from Step 2.
4) The final step was to test the algorithm with the best parameters with the same images using the post processing methods.

Figure 4 has the sample result of the detection process of the image with the post processing operation. The first row is the forgery image, the second line is the forgery mask, which shows the exact forgery area this mask is part from, and the last row is the result of the detection. The last row images have three different colors, each one representing a particular category:

- Sky color represents the correctly detected points, which means the points are in the mask as part of the forgery area and the detection process recognized the same.
- Green color represents the missing points, which are the points in the mask, but the detection process does not recognize them as part of the forgery area.
- Blue color represents the falsely detected points, which are points not in the mask but the process recognized them as part of the forgery area.

TABLE II: Post processing operation, tags and parameters

Operation	Tag	Parameters
JPEG Compression	JC1	Quality Factor = 20
	JC2	Quality Factor = 30
	JC3	Quality Factor = 40
	JC4	Quality Factor = 50
	. . .	. . .
	JC9	Quality Factor = 100
Image Blurring	IB1	Mean Value μ = 0, Variance σ 2 = 0.009
	IB2	Mean Value μ = 0, Variance σ 2 = 0.005
	IB3	Mean Value μ = 0, Variance σ 2 = 0.0005
Noise Adding	NA1	Average Filter = 3 3
	NA2	Average Filter = 5 5
	NA3	Average Filter = 7 7
Brightness Change	BC1	Brightness Ranges (0.01, 0.95)
	BC2	Brightness Ranges (0.01, 0.9)
	BC3	Brightness Ranges (0.01, 0.8)
Color Reduction	CR1	intensity levels per each color channel = 32
	CR2	intensity levels per each color channel = 64
	CR3	intensity levels per each color channel = 128
Contrast Adjustments	CA1	Adjustment Ranges = (0.01, 0.95)
	CA2	Adjustment Ranges = (0.01, 0.9)
	CA3	Adjustment Ranges = (0.01, 0.8)

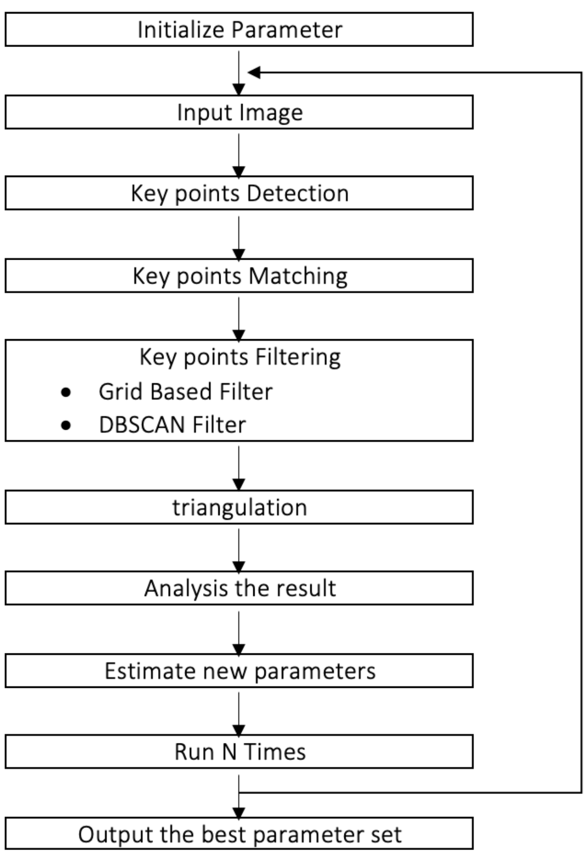


Fig. 3: Algorithm steps with optimization process

Each column shows the result for the image with one type of post processing action as follows:

- Column 1 image with Adjustment Ranges level 3;
- Column 2 image with Color Reduction level 3;
- Column 3 image with Noise Adding level 3;
- Column 4 image with Image Blurring level 1;
- Column 5 image with JPEG Compression level 9.

The details of these actions areas are given in Table II.

Figure 5 shows the average values of precision, recall and F1 for all experiments done using the optimized values. In all cases, the precision values are higher than the recall values since it measures the true points detected. The overall average of precision, recall and F1 are 76%, 61% and 65%, respectively. This result compared to other work to detect the forgery area is very good when comparing the results later on.

Figure 6 compares the precision, recall and F1 results between using optimal values and the default ones in this algorithm using the forgery images only without any other post processing action. The values are high since the editing process does not involve any Image blurring, JPEG compression, or other kind of modifications. There is a noticeable difference between the results in recall, which means less false detection points as a result of using better parameter values. The difference is around 10% improving the F1 results as well.

The remaining results are compared with

- TSF [1] a key point based algorithm with two filtering techniques;
- HFPM [9] a key point based algorithm with color features based filter;
- Zernike [17] a Block based algorithm;
- BusterNet [18], AR-NET [19] both are deep learning based algorithms;
- PatchMatch [23] a key point detection algorithm.

Figure 7 compares the F1 results for forgery detection using images with contrast adjustment. That attack remaps the image intensity values to the full display range of the data type. An image with good contrast has sharp differences between black and white. In the dataset, there are three ranges of contrast adjustment (0.01, 0.95), (0.01, 0.90) and (0.01, 0.80) and the result shows that the proposed method has surpassed all other methods in all three ranges. The result was around 0.77 in first

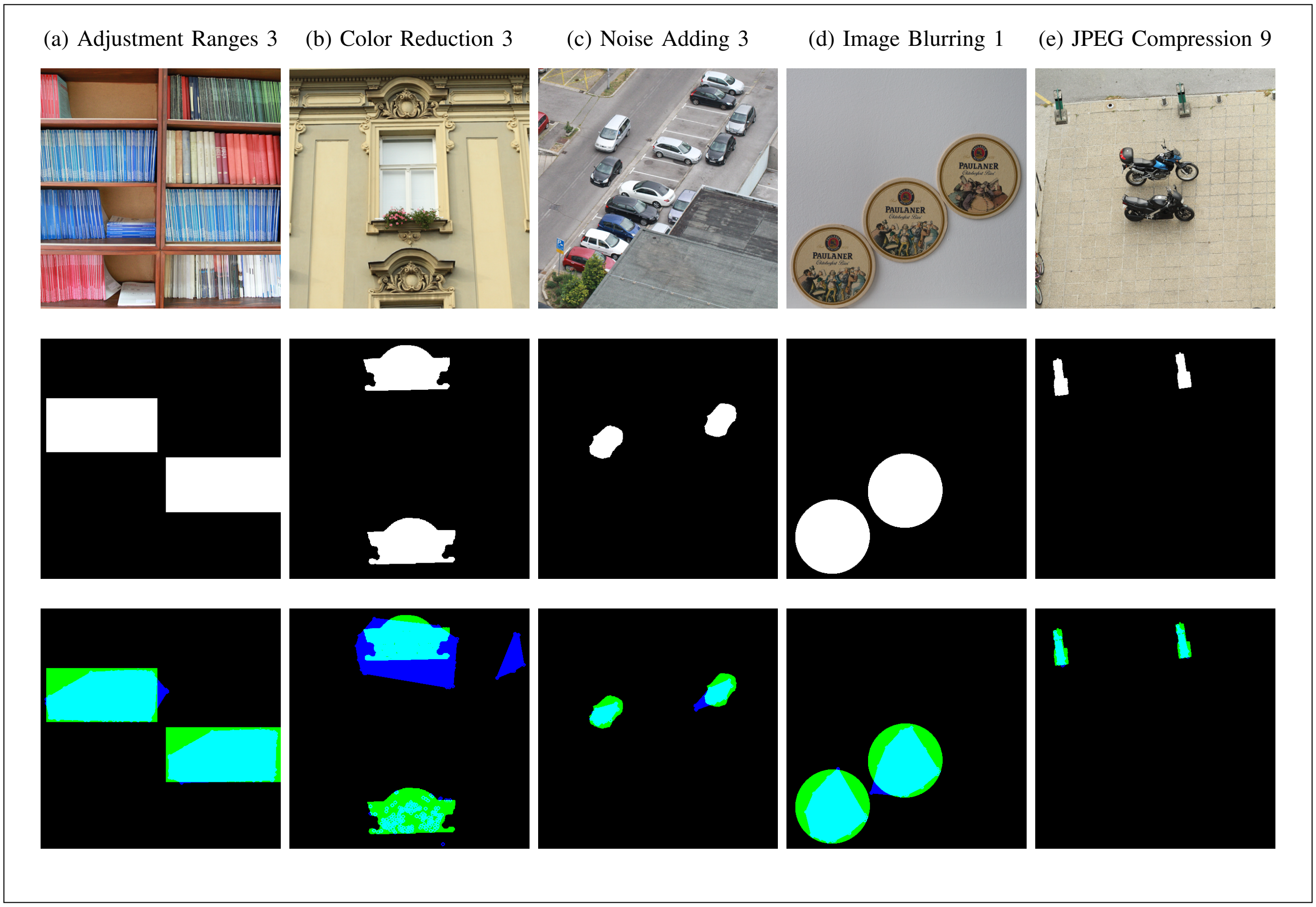


Fig. 4: Example of detection results

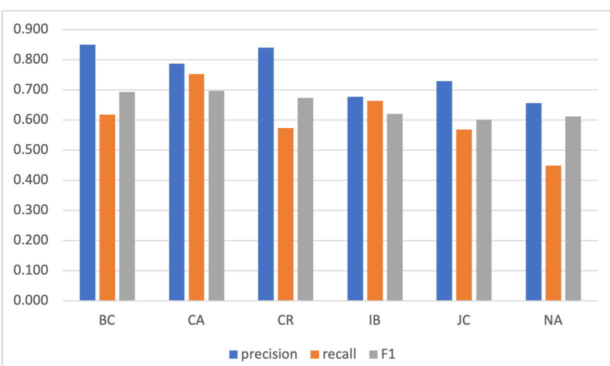


Fig. 5: Average values of precision, recall and F1

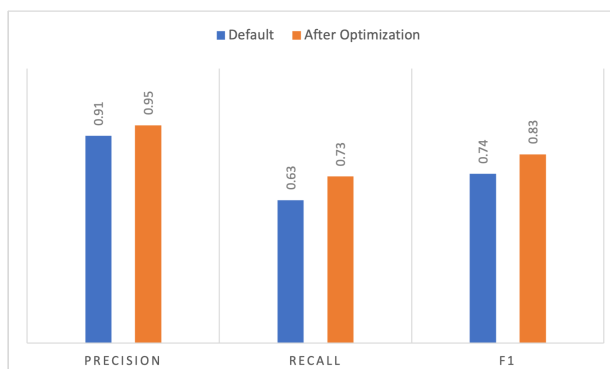


Fig. 6: Average values of precision, recall and F1

range, and around 0.76 in the other two. TSF achieves a good result with around 0.75 in all ranges.

Figure 8 compares the F1 values for images with color reduction post processing attack. There are also three levels from this attack with intensity levels per each color channel values as 32, 64 and 128. All method results were mostly equal for each level, for example AR-Net was 0.55, and TSF values between 0.65 and 0.70, but the HFPM method achieved a good result in the third value where the result increased from 0.55 to 0.70. However, our proposed algorithm obtains the best results with around 0.70 for all attack levels.

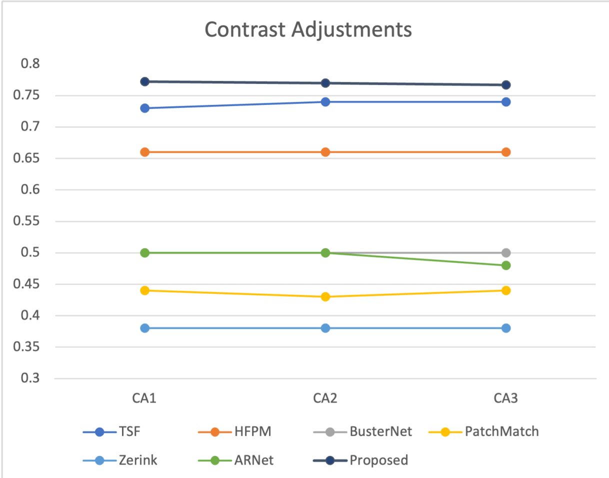


Fig. 7: F1 results for images with contrast adjustments

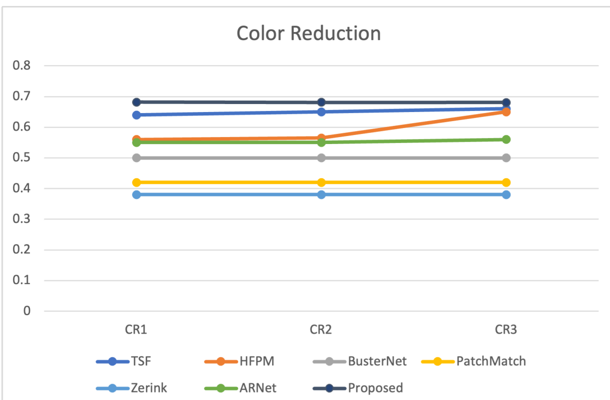


Fig. 8: F1 results for images with color reduction

Figure 9 shows the result of the testing image with blurring attack, which changes the images clearance to the negative side or makes it less distinct. The sigma value has been changed to three values, which are 0.009, 0.005, and 0.0005. Both TSF and proposed algorithm performance were very good surpassing the other algorithms for all attack levels.

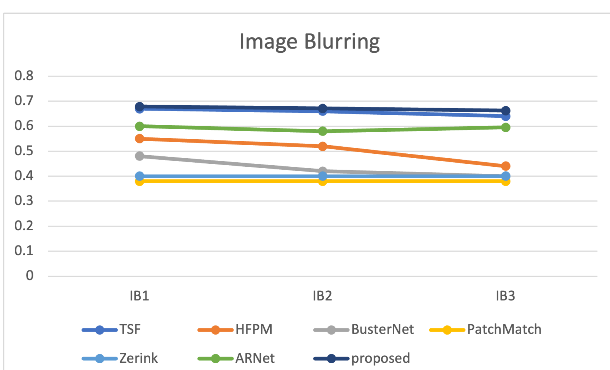


Fig. 9: F1 results for images with image blurring

Figure 10 is for image that changed with noise with 3 filter sizes. The first filter size was 3×3, TSF was the best algorithm with 0.54, ARNet was second with a value of 0.52, and our proposed algorithm achieving the third rank with around 0.45. The second filter size was 5×5 our proposed algorithm performance improved to 0.55, which was the same as for ARNet while TSF was the best with 0.60. In the third test with filter size 7×7, our proposed algorithm achieved better results than before and the average result was around 0.60, that is same as TSF. The ARNet result was constant at 0.52 for all three tests.

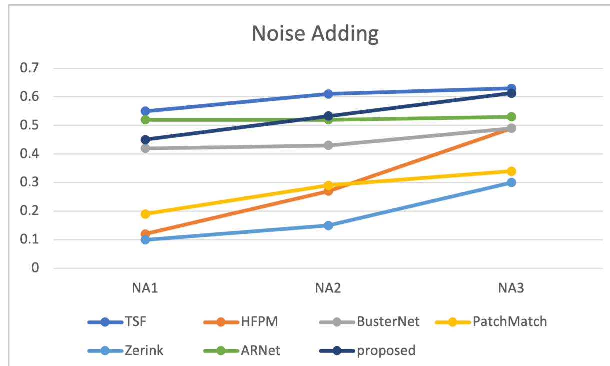


Fig. 10: F1 results for images with noise addition

Brightness Change is another attack applied to forgery images, this attack adds or removes constant values from all pixels' original values in the image. Figure 11 shows the result with three level of change. Our proposed algorithm outperforms all others with values around 0.70 for all cases.

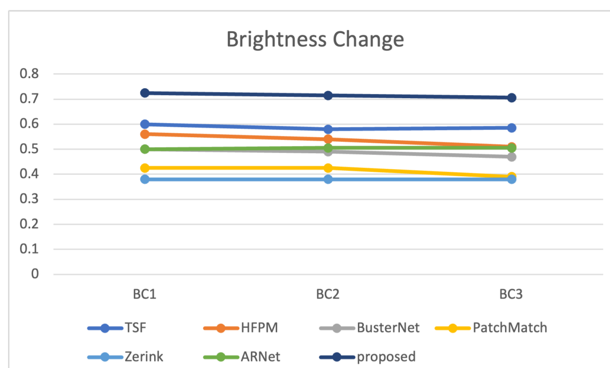


Fig. 11: F1 results for images with brightness change

The last attack that was applied to images was JPEG Compression. That attack reduces the size of the image without affecting its quality. The attack was applied with reduction quality factor between 20 and 100. Figure 12 shows the results for the algorithms for nine different quality factors. The results indicates that TSF was the best algorithm for the first 4 quality factors, however, our proposed algorithm performs as as well as TSF for the fifth factor and after that it surpasses all other algorithms. The F1 values achieved by our proposed algorithm were between 0.58 to 0.70, while the TSF values were 0.58

to 0.65. ARNet was performing good as well with F1 values range 0.5 and 0.6.

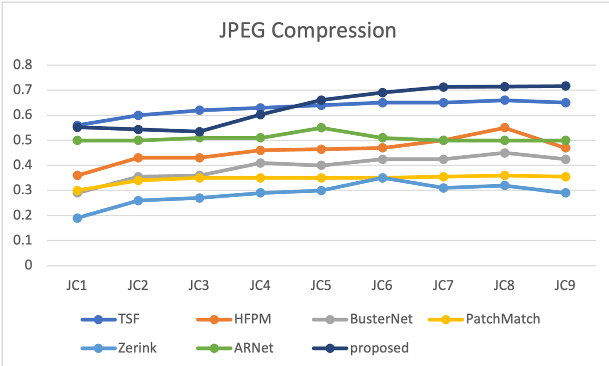


Fig. 12: F1 results for images with JPEG compression

V. Conclusion

This paper proposed a new algorithm to detect image forgery with move and copy attacks. For this type of attack a part of the image is copied and pasted to another region of the image. This copy move and paste process can include geometric operations such as scaling, rotating, reflecting, translating, and affine transformation of the copied area. Further, the post processing operation is another type of operation applied while image editing, that includes JPEG compression, adding Gaussian noise, color reduction, contrast adjustment, brightness change, and image blurring. The new algorithm was composed of key points detection, matching filtering using grid based filter and DBSCAN, and triangulation. In each phase of the proposed method, there are parameters that have to be configured to make this algorithm reliable and powerful. Eight parameters were identified in all phases, which are the number of octave layer, contrast threshold, edge threshold, sigma, matching thresh hold block size, Eps, and minimum samples. PSO was used to find the best values for each of these parameters through all images of the CoMoFoD dataset. The new algorithm has been tested with optimized values achieved and was compared to other six algorithms. The results show that the proposed surpasses the other algorithms in 18 of 24 test. The algorithm was also compared using the default parameter and optimal values.

VI. Acknowledgment

This work used resources of the Center for Computationally Assisted Science and Technology (CCAST) at North Dakota State University which was made possible in part by the NSF Award No. 2019077.

References

[1] J. Yang, Z. Liang, Y. Gan, and J. Zhong, A novel copy-move forgery detection algorithm via two-stage filtering, Digital Signal Processing, vol. 113. Elsevier BV, p. 103032, Jun. 2021. doi: 10.1016/j.dsp.2021.103032.

[2] E. Rosten and T. Drummond, Fusing points and lines for high performance tracking, in Proceedings of the IEEE International Conference on Computer Vision, pp. 15081511, IEEE, Beijing, China, October 2015.

[3] S. Leutenegger, M. Chli, and R. Y. Siegwart, BRISK: binary robust invariant scalable keypoints, in Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 25482555, IEEE, Barcelona, Spain, November 2011.

[4] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, ORB: an efficient alternative to SIFT or SURF, in Proceedings of the 2011 International Conference on Computer Vision, pp. 25642571, IEEE, Barcelona, Spain, November 2011.

[5] H. Bay, T. Tinne, and V. G. Luc, SURF: speeded up robust features, Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346359, 2008.

[6] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision, vol. 60, no. 2, pp. 91110, 2004.

[7] P. F. Alcantarilla, A. Bartoli and A. J. Davison, KAZE Features, ECCV 2012, vol. 7577, Springer, London, UK, 2012.

[8] S.F. Hajialilu, M. Azghani, N. Kazemi, Image copy-move forgery detection us- ing sparse recovery and keypoint matching, IET Image Process. 14 (12) (2020) 2799-2807.

[9] Y. Li and J. Zhou, Fast and Effective Image Copy-Move Forgery Detection via Hierarchical Feature Point Matching, in IEEE Transactions on Information Forensics and Security, vol. 14, no. 5, pp. 1307-1322, May 2019, doi: 10.1109/TIFS.2018.2876837.

[10] F. Zhao, R. Zhang, H. Guo and Y. Zhang, Effective digital image copy-move location algorithm robust to geometric transformations, 2015 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), 2015, pp. 1-5, doi: 10.1109/ICSPCC.2015.7338958.

[11] A.J. Fridrich, B.D. Soukal, A.J. Luk, Detection of copy-move forgery in digital images, in: Proc. Digital Forensic Research Workshop, 2003.

[12] U. Gani, F.Qadir, A robust copy-move forgery detection technique based on discrete cosine transform and cellular automata, Journal of Information Security and Applications, Volume 54, 2020, 102510, ISSN 2214-2126, https://doi.org/10.1016/j.jisa.2020.102510.

[13] G. Muhammad, M. Hussain, G. Bebis, Passive copy move image forgery detec- tion using undecimated dyadic wavelet transform, Digit. Investig. 9 (1) (2012) 49-57.

[14] Y. Li, Image copy-move forgery detection based on polar cosine transform and approximate nearest neighbor searching, Forensic Sci. Int. 224 (1) (2013) 59-67.

[15] Y. Wang, X. Kang and, Y. Chen, Robust and accurate detection of image copy- move forgery using PCET-SVD and histogram of block similarity measures, J. Inf. Secur. Appl. 54 (2020) 102536.

[16] J. Deng, J. Yang, S. Weng, G. Gu, Z. Li, Copy-move forgery detection robust to various transformation and degradation attacks, KSII Trans. Int. Inf. Syst. 12 (9) (2018) 44674486.

[17] R.Seung-Jin & L. Min-Jeong & L Heung-Kyu. (2010). Detection of Copy-Rotate-Move Forgery Using Zernike Moments. LNCS. 6387.

[18] Y. Wu, W. Abd-Almageed, P. Natarajan (2018). BusterNet: Detecting Copy-Move Image Forgery with Source/Target Localization. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds) Computer Vision ECCV 2018. ECCV 2018. Lecture Notes in Computer Science(), vol 11210. Springer, Cham. https://doi.org/10.1007/978-3-030-01231-1_11

[19] Y. Zhu, C. Chen, G. Yan, Y. Guo and Y. Dong, AR-Net: Adaptive Attention and Residual Refinement Network for Copy-Move Forgery Detection, in IEEE Transactions on Industrial Informatics, vol. 16, no. 10, pp. 6714-6723, Oct. 2020, doi: 10.1109/TII.2020.2982705.

[20] D. G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, vol. 60, no. 2. Springer Science and Business Media LLC, pp. 91110, Nov. 2004. doi: 10.1023/b:visi.0000029664.99615.94.

[21] S. Wenchang, Z. Fei, Q. Bo and L. Bin, Improving image copy-move forgery detection with particle swarm optimization techniques, in China Communications, vol. 13, no. 1, pp. 139-149, Jan. 2016, doi: 10.1109/CC.2016.7405711.

[22] D. Tralic, I. Zupancic, S. Grgic, M. Grgic, CoMoFoD - New Database for Copy-Move Forgery Detection, in Proc. 55th International Symposium ELMAR-2013, pp. 49-54, September 2013.

[23] D. Cozzolino, G. Poggi and L. Verdoliva, Efficient Dense-Field Copy-Move Forgery Detection, in IEEE Transactions on Information Forensics and Security, vol. 10, no. 11, pp. 2284-2297, Nov. 2015, doi: 10.1109/TIFS.2015.2455334.