Digital Media Forgery Detection

BY

MOHAMMED ALORAINI
BSc, Qassim University, 2011
M.S., University of Illinois at Chicago, 2014

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Chicago, 2020

Chicago, Illinois

Defense Committee:

Dan Schonfeld, Chair and Advisor
Rashid Ansari
Mojtaba Soltanalian
Ahmet Enis Cetin
Brian Ziebart, Computer Science

To my father, Ibrahim, my mother, Khadijah, and my family

for their unconditional love and support

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my Ph.D. advisor, Prof. Dan Schonfeld, for his continuous guidance, inspiration, dedication, and encouragement. I am truly grateful and honored to work with him during the past years. This thesis would not have been possible without his supervision and endless support.

I would also like to express my sincere appreciation to my dissertation committee, Prof. Rashid Ansari, Prof. Mojtaba Soltanalian, Prof. Ahmet Enis Cetin, and Prof. Brian Ziebart, for spending their valuable time serving on the committee. I am thankful for their discussions and suggestions throughout this journey. I also want to thank my labmates for constructive collaboration that contributed to this thesis.

Finally, I would like to thank my beloved parents, Ibrahim and Khadijah, and my family for their unconditional love, inspiration, and encouragement. This journey would not have been possible without the dedicated support from them.

MA

CONTRIBUTION OF AUTHORS

Chapter 1 presents the thesis introduction and summarizes research contributions.

Chapter 2 provides the related works of detecting image and video forgeries.

Chapter 3 presents a published paper (Aloraini et al., 2019 [1]), for which I was the first author and main contributor. Dr. Lingdao Sha and Dr. Mehdi Sharifzadeh contributed to developing the idea and editing the manuscript. Prof. Dan Schonfeld was the lead investigator.

Chapter 4 presents two published papers (Aloraini et al., 2019 [2], and Aloraini et al., 2020 [3]), for which I was the first author and main contributor. Dr. Mehdi Sharifzadeh and Chirag Agarwal contributed to developing the idea and editing the manuscript. Prof. Dan Schonfeld was the lead investigator.

Chapter 5 presents a paper that was submitted to IEEE Transactions on Circuits and Systems for Video Technology. I was the first author and main contributor. Prof. Dan Schonfeld was the lead investigator.

Chapter 6 concludes this thesis and discusses possible future works.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF TABLES (Continued)

TABLE **PAGE**

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

SUMMARY

In recent years, digital media have been used as indisputable evidence of a crime, and it is therefore important to ensure the reliability of these digital media. Unfortunately, digital media can be forged by using the advent of powerful and easy-to-use media editing tools. A forged digital medium is often eye-deceiving and appears in a way that is realistic, hence believable. A fundamental challenge is to determine whether a digital medium is authentic or not. This task is particularly challenging due to the lack of ground truth bases that can be used to verify the originality and integrity of digital media content. In this thesis, we investigate digital media forgery, particularly image and video forgery, and propose novel approaches to detect and localize digital media forgery.

The first part of the thesis is devoted to image forgery detection. We propose a novel approach that uses dictionary learning and sparse coding to detect digital image forgery. We also propose a new matching criterion that is performed using dictionary atoms instead of traditional matching criteria. We conduct our experiments using two popular data sets to determine how effectively and efficiently our approach detects digital image forgery compared to previous approaches. The experimental results show that our approach outperforms state-of-the-art approaches and leads to robust results against compression and rotation attacks. Furthermore, our approach detects forgery significantly faster than these approaches since it uses a sparse representation that dramatically reduces dimensions of feature vectors.

The second part of the thesis is devoted to object removal video forgery detection. We propose a novel approach based on sequential and patch analyses to detect object removal forgery and to localize forged regions in videos. Sequential analysis is performed by modeling video sequences as stochas-

tic processes, where changes in the parameters of these processes are used to detect a video forgery. Patch analysis is performed by modeling video sequences as a mixture model of normal and anomalous patches, with the aim to separate these patches by identifying the distribution of each patch. We localize forged regions by visualizing the movement of removed objects using anomalous patches. We conduct our experiments at both pixel and video levels to determine the effectiveness and efficiency of our approach to detection of video forgery. The experimental results show that our approach achieves excellent detection performance with low-computational complexity and leads to robust results for compressed and low-resolution videos.

The third part of the thesis is devoted to facial manipulation detection. We propose a novel approach, dubbed FaceMD, based on fusing three streams of convolutional neural networks to detect facial manipulation. The proposed FaceMD incorporates spatiotemporal information by fusing video frames, motion residuals, and 3D gradients to improve facial manipulation detection accuracy. We combine these three streams using different fusion methods and places to best use this spatiotemporal information, hence increasing detection performance. The experimental results show that the proposed FaceMD achieves state-of-the-art accuracy using two different facial manipulation data sets.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Recently, digital media have become essential for security applications used to monitor many organizations and locations. Many legal institutions have also used digital media as legal evidence for judgments. Therefore it is important to ensure the reliability of these digital media. Unfortunately, digital media can be forged by using the advent of powerful and easy-to-use media editing tools. If a digital medium is manipulated, it could lead to many critical problems that are related to public security or legal evidence. This manipulated digital medium is often eye-deceiving and appears in a way that is realistic and believable. Media are sometimes tricked and used fake digital media as if they are real. As a result, digital media should be carefully analyzed to ensure their originality and integrity.

In general, digital media forgery can be divided into two categories: image and video forgeries. Image forgery can also be divided into two categories: image splicing and image copy-move forgeries. Image splicing is a process of combining regions from two or more images to form a forged image [4,5]. Image copy-move forgery is created by copying a part of an image and pasting this part in another region of the same image [6]. This copy-move forgery is more common than splicing forgery because a forger uses only one image to make a forgery.

Video forgery can be divided into three categories: frame-based forgery, object-based forgery, and facial manipulation [7]. Frame-based forgery is created by deleting frames from a video scene, inserting

frames into the video scene, or duplicating frames in the video scene. This forgery is easy to perform by using any of the basic editing tools because a manipulator needs only to divide a video into frames to create a video forgery. Object-based forgery is created by adding new moving objects to a video scene or removing existing moving objects from the video scene. It is difficult to add moving objects without leaving invisible traces since videos might expose different motions and illuminations. Hence, object-based video forgery often refers to removing objects from a video. This forgery is more complicated to perform compared to frame-based forgery because a forger needs to manipulate specific regions in video frames while maintaining temporal consistency between these frames.

Facial manipulation can be divided into two categories: facial identity manipulation and facial expression manipulation. Facial identity manipulation is created by replacing a person's face with another person's face. The most common facial identity manipulation approaches are FaceSwap [8] and DeepFakes [9]. Facial expression manipulation is created by replacing a person's expressions with another person's expressions while maintaining facial identity. The most common facial expression manipulation approaches are Face2Face [10] and NeuralTextures [11]. FaceSwap and Face2Face approaches are based on computer graphics techniques, whereas the DeepFakes and NeuralTextures approaches are based on deep learning techniques.

Creating an automatic approach to detect digital media is a very difficult problem due to the lack of truthful bases that can be used to verify the originality and integrity of digital media contents. A forged digital medium may not only be attacked by one of digital media forgeries, but also could be attacked by other complex processes, including compression, resizing, and rotation. These processes make forgery detection more challenging. Furthermore, if a forger removes an object (e.g., person)

from a video scene, it becomes difficult to detect forged regions due to the high correlation between these forged regions and original regions. As a result, it is challenging to ensure the originality and integrity of digital media content.

1.2 Research Contributions

We summarize the main contributions of this thesis as follows:

1. We propose a novel approach that uses dictionary learning and sparse coding to detect digital image forgery (Aloraini et al., 2019 [1]).
2. We propose a novel approach based on sequential and patch analyses to detect object removal forgery and localize forged regions in videos (Aloraini et al., 2019 [2], and Aloraini et al., 2020 [3]).
3. We propose a novel approach, dubbed FaceMD, based on fusing three streams of convolutional neural networks to detect facial manipulation. This work is submitted to IEEE Transactions on Circuits and Systems for Video Technology.

1.3 Thesis Outline

The rest of the thesis is organized as follows: In Chapter 2, we provide related works of detecting image and video forgeries. Chapter 3 presents dictionary learning and sparse coding for digital image forgery detection. Chapter 4 presents sequential and patch analyses for object removal video forgery detection and localization. Chapter 5 presents FaceMD: convolutional neural network-based spatiotemporal fusion facial manipulation detection. Chapter 6 concludes this thesis and discusses possible future works.

CHAPTER 2

RELATED WORKS

2.1 Image Forgery Detection

Most of the existing image splicing forgery detection approaches can be divided into four categories: illumination-based, noise-based, camera-based, and feature-based. The illumination-based approaches use inconsistencies of an illuminant color that is estimated from different regions of an image as evidence of unoriginality [12, 13]. The noise-based approaches use local noise inconsistencies as an indication of image forgery [14–16]. The camera-based approaches use artifacts that are generated from using different camera lenses and sensors to detect forgery [17, 18]. The feature-based approaches start by extracting specific features from an image and then use a classifier to classify between two classes: authentic vs. forged [19, 20].

Current existing approaches for image copy-move forgery detection can be divided into two categories: block-based and keypoint-based approaches [21]. First, block-based approaches divide an image into patches and then detect forgery by looking for similar patches. The representative approaches are DCT [22], PCA [23], DWT with KPCA [24], and Zernike moment [25]. Second, keypoint-based approaches begin with extracting interest points (keypoints), such as edges and corners from an image, and then finding similarities between these points [26]. The representative approaches are SIFT [27, 28] and SURF [29].

Other works combine block-based with keypoint-based approaches to enhance detection results [30–32]. Jian Li et al. introduced an expectation-maximization (EM) stage after segmenting an image into patches to reduce transform estimation error between copy and original areas [30]. Although this stage improves detection results, it imposes a high computational cost because of the EM algorithm's iterative procedure. Ferreira et al. combined different detector approaches and extended behavior knowledge space representation fusion, to enhance detection accuracy [32]. However, this approach is computationally expensive since it combines many detection approaches.

2.2 Video Forgery Detection

We divide this section into two parts. We first discuss related works of detecting frame-based and object-based forgeries. Then, we present related works of facial manipulation and facial manipulation detection.

2.2.1 Detection of Frame-Based and Object-Based Forgeries

Although several works have been conducted to review video forensic approaches [33–37], most of these works focused on detecting frame-based forgery [38]. These works can be divided into three categories: motion-based [39–43], correlation-based [44–47], and compression-based [48–52]. First, motion-based approaches use inconsistencies of motion vectors as an evidence of a frame deletion or insertion. A drawback of these approaches is that the detection accuracy decreases when compression increases. Second, correlation-based approaches use high correlation between suspicious frames as an indication of a frame duplication. These approaches fail to detect the frame duplication when the frame duplication occurs in static background frames or performs in a different order. Third, compression-based approaches declare video forgery by detecting double compression. These approaches are not

applicable when a complete group of pictures (GOP) is removed, or recompression is occurred without video tampering.

Only a few works have been conducted to detect object-based forgery compared to frame-based forgery [53]. These works tackle two types of object-based forgery: object insertion video forgery and object removal video forgery. The following works are proposed to detect object insertion video forgery [54–60]. Some approaches use correlation between blurring features [54], or edge features [55], to detect blue screen compositing. The forgery is detected by examining changes in correlation patterns between these features. These approaches fail to detect video forgery if the background of a video is green or blue. Other approaches use DCT coefficients [56], or luminance and contrast [58], as local features to measure the similarity between foreground and background. The forgery is detected by identifying inconsistencies in these features between foreground and background. A limitation of these approaches is that the detection accuracy decreases when the bit rate of videos decreases. Conotter et al. proposed an approach that uses projectile motion to identify falsified objects [59]. D'Avino et al. presented an approach that uses deep learning to learn an intrinsic model of an original video, where a video is classified as forged if it does not fit the learned model [60].

Object removal video forgery is achieved by using inpainting algorithms [61–63]. The following works are proposed to detect object removal video forgery [64–72]. Zhang et al. developed an approach that uses ghost shadow artifact to identify inconsistencies between foreground mosaic and trajectory of moving foreground [64]. Hsu et al. introduced an approach that uses temporal correlation of noise residues to identify irregular changes in the correlation of noise residues throughout video frames [65]. A similar approach uses correlation of Hessian matrices to detect object removal forgery [67]. Richao

et al. presented an approach that uses object contour features with a support vector machine (SVM) algorithm to detect removed moving objects with static background [69]. Another approach uses steganalytic features, which are extracted from motion residual matrices, with ensemble classifiers to classify a frame into three categories: pristine, forged, and double compressed [70]. Lichao et al presented an approach based on compressive sensing to detect removed moving objects with static background [71]. All of the above works can detect video forgery, but they cannot localize forged regions in videos. Lin et al. introduced an approach based on spatiotemporal coherence analysis to detect and localize tampered regions [72]. A limitation of this approach is that detection performance drops significantly when tampered videos are saved in compressed formats. Deep Convolutional Neural Networks (CNNs) require large data sets for training to achieve excellent results [37]. However, there are a few object removal forgery data sets that are publicly available [36, 37]. As a result, CNNs are not ideal to tackle the object removal forgery problem.

2.2.2 Facial Manipulation and Facial Manipulation Detection

Facial manipulation approaches are divided into two categories: computer graphics-based and learning-based approaches [73–75]. Computer graphics-based approaches reconstruct and track a 3D model of source and target faces, and transfer identity or expressions from the target face to the source face [76]. The following works are based on computer graphics to create facial manipulation videos [10, 77–79]. Dale et al. used a 3D multilinear model to create face swap manipulation by replacing the source face with the target face [77]. Garrido et al. proposed a 2D warping strategy to swap facial identity while keeping the original expressions [78]. Thies et al. introduced the first real-time facial expression ma-

nipulation using a consumer-level RGB-D camera [79]. Thies et al. proposed Face2Face that creates an advanced real-time facial reenactment of monocular videos [10].

Learning-based approaches track source and target faces and then use deep convolutional neural networks to swap facial identity or expression [74]. The following works are based on deep learning [9, 11, 80, 81]. DeepFakes introduced using two autoencoders with a shared encoder to create facial identity manipulation videos [9]. Doublicat is a similar approach that uses Generative Adversarial Network (GAN) to perform face swap manipulation [80]. Kim et al. proposed an approach that uses a generative neural network with a space-time architecture to replace a 3D head position, face expression, and eye blinking [81]. Thies et al. developed NeuralTextures that uses a rendering network with a U-Net architecture to create facial expression manipulation [11].

Computer graphics-based and learning-based approaches create independent facial manipulation frames, hence this facial manipulation causes spatiotemporal artifacts. In particular, these approaches start by dividing source and target videos into sequences of video frames. Then, they transfer facial identity or expression from each frame in the source video to the corresponding frame in the target video. Finally, they combine these frames to form a facial manipulation video. As a result, this video manipulation would cause temporal inconsistency between adjacent frames because video frames are generated independently. Furthermore, This video manipulation could produce edges at unexpected edge regions due to different sizes of different faces, or it couldn't produce edges at expected edge regions due to blurring artifacts. Therefore, this video manipulation causes spatial artifacts in addition to temporal artifacts.

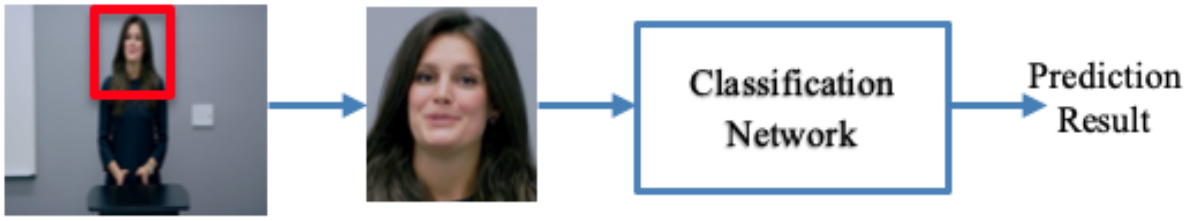


Figure 1: Current facial manipulation detection pipeline.

Some facial manipulation detection approaches use specific artifacts as an indication for manipulation [13, 82–88]. Agarwal et al. proposed an approach that uses facial action units to detect deepfake manipulation of world leaders [82]. Yang et al. developed an approach that uses inconsistencies in a 3D head pose estimation to detect face swap manipulation [83]. Tien et al. introduced an approach that uses facial expressions variation to detect computer-generated faces [84]. Carvalho et al. used inconsistencies in the illumination map to detect facial identity manipulation [13, 85]. Li et al. used eye blinking to detect face swap manipulation and showed that the first generation of fake faces either didn't blink or didn't blink at the expected frequency [88]. All of these works are now less effective because these works detect specific artifacts that have been solved in the next generation of facial manipulation approaches.

Other facial manipulation detection approaches propose a deep neural network that uses general artifacts as an indication for manipulation [89–96]. Current facial manipulation detection approaches follow the detection pipeline that is shown in Fig.1. These approaches start by tracking an input face, extracting face region, followed by a learned classification network that predicts whether the input face

is original or fake. Afchar et al. proposed MesoNet that is inspired by Inception [97] to detect both identity and expression manipulation [89]. David et al. combined a convolutional neural network with a Long Short Term Memory network to detect deepfake manipulation [90]. Zhou et al. proposed two-stream neural networks that only use spatial information by combining face classification stream with patch triplets stream to ensure patches of an image are close in the embedding space [91]. Raghavendra et al. used VGG19 and AlexNet networks to detect morphed face images [92]. Li et al. introduced a network to detect wrapping artifacts of deepfake manipulation [95]. Stehouwer et al. used a deep convolutional neural network with an attention map to detect facial manipulation [96]. All of these works are not robust against simple attacks such as additive noise, compression, and resizing [76]. However, these attacks are common in real-world scenarios; for example, a shared video in social media is usually compressed.

CHAPTER 3

DICTIONARY LEARNING AND SPARSE CODING FOR DIGITAL IMAGE FORGERY DETECTION

In this chapter, we investigate the image copy-move forgery problem, i.e., a part of the image is copied and pasted in another part of the same image to add or hide objects. Two examples of image copy-move forgery are illustrated in Fig.2. We also investigate the computational cost and propose a novel approach that is based on a sparse representation of keypoint descriptors to reduce the dimensionality of these descriptors and to remove noisy features from them. We utilize sparse coding,i.e., an unsupervised algorithm aim to learn set of overcomplete basis vectors (atoms) to represent data efficiently. We use sparse coding instead of traditional dimension reduction techniques such as PCA for two main reasons. *First*, sparse coding is able to learn overcomplete atoms and doesn't require these atoms to be orthogonal. *Second*, sparse coding has been widely used in image classification and pattern recognition and it has achieved promising performance. Thus, it is suitable for image forgery problem since it is binary classification, i.e., tampered vs. authentic image.

3.1 Contributions

The contributions of this chapter can be summarized as follows:

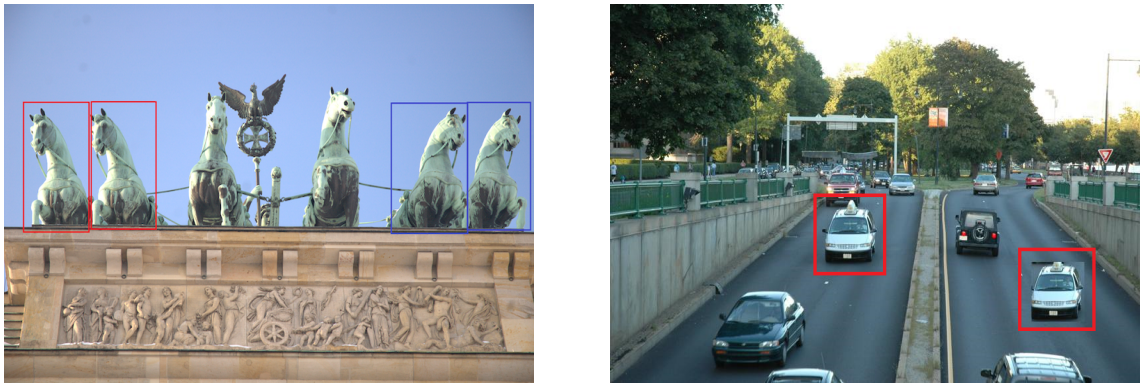


Figure 2: Two examples of image copy-move forgery. Two horses are cloned on the left image and one car is cloned on the right image. Cloned sections are squared in different colors.

1. We propose a novel matching criterion based on dictionary atoms that results in a more effective and efficient forgery detection approach when compared to the original SIFT matching model [27].

2. Our approach is robust against compressions and rotations attacks since it uses sparse representations that better fit features' descriptors of an image.

3. Our approach is scalable since its computational complexity is significantly reduced by using low-dimensional feature vectors of an image.

The rest of this chapter is organized as follows: Our proposed approach is described in Sec. 3.2. Feature extraction is presented in Sec. 3.2.1. Sparse coding is presented in Sec. 3.2.2. The sparse matching is explained in Sec. 3.2.3. Geometric transformation estimation is presented in Sec. 3.2.4. The experimental results are provided in Sec. 3.3.

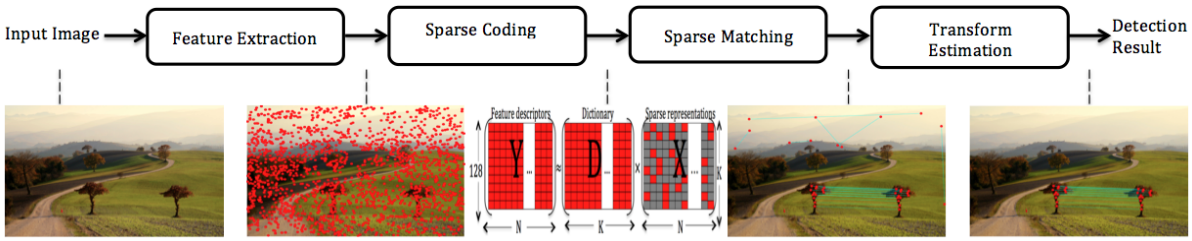


Figure 3: Flowchart of the proposed approach.

3.2 Image Copy-Move Forgery Detection

We briefly describe our approach in the following steps, as illustrated in Fig.3. First, we extract Scale Invariant Feature Transform (SIFT) [98] from an image. Second, K means-Singular Value Decomposition algorithm (K- SVD) [99] is utilized to obtain a sparse representation of SIFT descriptors. Third, the matching process is performed by finding similarities between these sparse features. Next, agglomerative hierarchical clustering [100] is applied on spatial locations of the matched points to identify possible cloned areas. Finally, geometric transformation estimation is obtained between these cloned areas by using RANdom SAmple Consensus algorithm (RANSAC) [101]. An image is forged if a uniform transformation matrix can be obtained between any two matched areas.

3.2.1 Features Extraction

A scale invariant feature transform algorithm (SIFT) [98] is used for keypoint detection and description since SIFT is more robust against scaling, rotating, and illumination. The SIFT algorithm starts with generating 4 octaves (i.e., images with the same size) each of which has 5 images with 5 different blur levels(scales). In each octave, any two consecutive images are subtracted to obtain Difference

of Gaussian (DoG) images. Then, maxima and minima (i.e., keypoints) in DoG images are detected by comparing neighboring pixels in the same scale and neighboring scales. If any keypoint has an intensity below a predefined threshold or lies along an edge, it is rejected. Subsequently, a 16×16 window is taken around the keypoint and broken into a 4×4 window. Then, gradient magnitudes and orientations are calculated to generate an 8-bin histogram, which is used to form $128(4 \times 4 \times 8)$ elements as a feature vector,i.e., descriptor.

3.2.2 Sparse Coding

Sparse representation is a method of representing data via a linear combination of dictionary atoms (columns). Given $Y \in R^{128 \times N}$ as SIFT descriptors of an image,the goal is to find a dictionary with K atoms $D \in R^{128 \times K}$ and a representation $X \in R^{K \times N}$ such that $Y \approx DX$ and X is sparse enough (3.1). We use an adaptive dictionary learning method called K-SVD [99] that has two stages: sparse coding and dictionary update. First, the K-SVD starts with initializing random dictionary D. Then, during sparse coding stage, it finds the best sparse representations X using an orthogonal matching pursuit algorithm [102], given the current dictionary D. Next, during dictionary update stage, it updates dictionary atoms one at a time by using the current sparse representations X. Then, it iterates until the algorithm converges or reaches a predefined number of iterations.

$$\min_{D,X} \{ \|Y - DX\|_F^2 \} \quad \text{s.t.} \quad \forall i, \|x_i\|_0 \leq S \tag{3.1}$$

F denotes the Frobenius norm, and $\| \cdot \|_0$ is the $\mathscr{L}^0$ pseudo-norm that counts the non-zero entries.

By using the K-SVD algorithm, we approximate SIFT features (128 elements) based on just 6 dictionary atoms

3.2.3 Sparse Matching

We have experimentally observed that similar keypoints tend to use the same dictionary atoms in their sparse representations but with different sparse coefficients. For this reason, we propose a novel matching criterion to detect multiple copies of the same features (keypoints), where a keypoint matches other keypoints if their sparse representations are obtained by using the same dictionary atoms. In other words, if a_i is a vector that locates non zero entries in $x_i \in R^K$ which is a sparse representation of a keypoint descriptor $y_i \in R^{128}$ for a keypoint i, then the keypoint i matches another keypoint j if and only if $a_i = a_j$. We obtain the set of matched keypoints by iterating over sparse representations of the keypoints descriptors in an image.

3.2.4 Geometric Transformation Estimation

Given the matched keypoints in an image, we employ agglomerative hierarchical clustering [100] on spatial locations of the keypoints. Hierarchical clustering begins with one keypoint in each cluster, then it combines the closest pair of clusters into a single cluster, and computes the distances between the new cluster and all the other clusters. The clustering process is repeated until a threshold condition is reached to segregate original regions from copy regions. After clustering is performed, we estimate the affine transformation matrix H between any pair of matched clusters. Let x_i and x_i' be the homogenous coordinates of the matched keypoints in the copy region and original region, respectively. Then the geometric relationship between them is defined as follows:

$$x_i' = Hx_i \tag{3.2}$$

Considering the existence of outliers (mismatched keypoints), we perform matrix estimation using the RANSAC algorithm [101]. The algorithm estimates the matrix H by randomly selecting three matched pairs, and then it transforms all other points using H. A pair of matched keypoints is an inlier if the distance between the keypoint and the corresponding transformed one is less than a predefined threshold. The process is repeated until a predefined number of iterations is achieved. Finally, the estimated matrix H, which results in a large number of inliers, is elected.

3.3 Experimental results

We conduct our experiments on two public data sets. The first data set is the Image Manipulation Data set (IMD) [21], which consists of 48 original images, 48 plain CMF images, and 1392 images that have a single attack, i.e., rotation or noise addition, or JPEG compression. The second data set is MICC-F600 [27], which contains 440 original images and 160 forged images. The 160 forged images consist of 40 images that have one duplicated region, 40 images that have two or three duplicated regions, 40 images that have one duplicated region with 30° rotations, and 40 images that have one duplicated region with 30° rotations and 120% scaling. We have chosen these two data sets because they are used in the validation of a recent work [30] and according to Amerini et al. [27], the MICC-F600 data set is the most challenging data set among the other data sets that were constructed by them.

3.3.1 Evaluation Metric

By defining T_P as the correctly detected forged images, F_P as original images that have been incorrectly detected as forged and F_N as falsely missed forged images, we compute *Precision*, *Recall*, and $F1$ as follows:

$$Precision = \frac{T_P}{T_P + F_P} \tag{3.3}$$

$$Recall = \frac{T_P}{T_P + F_N} \tag{3.4}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Presision + Recall} \tag{3.5}$$

Precision shows the probability that a detected forgery is truly a forgery, *Recall* indicates the probability that a forged image is detected, and $F1$ score combines precision and recall in a single value.

3.3.2 Comparison of Detection Results

We compare our approach with three different approaches on the two data sets. One recent work, Segmented SIFT [30], which was introduced briefly in the first section, in addition to G2NN-SIFT [27], which handles multiple matches using g2NN, are selected for comparison. We also impliment PCA-SIFT [103] that reduces the dimension of the feature vector to 20 elements. We have chosen these approaches to compare with a recent work, SIFT-based approach, and dimensionality reduction approach. All these approaches including the proposed approach are implemented on a machine with an *Intel Core i7 with 8-GB RAM*. Readers are referred to [104] for more details about our implantation and source code.

3.3.3 Sparsity Settings

We randomly choose 10% of the data sets to tune the sparsity parameter of our test to achieve the lowest possible feature dimension that leads to high performance. We empirically choose the dictionary with 512 atoms. The receiver operating characteristic curve that is illustrated in Fig.4 suggested that

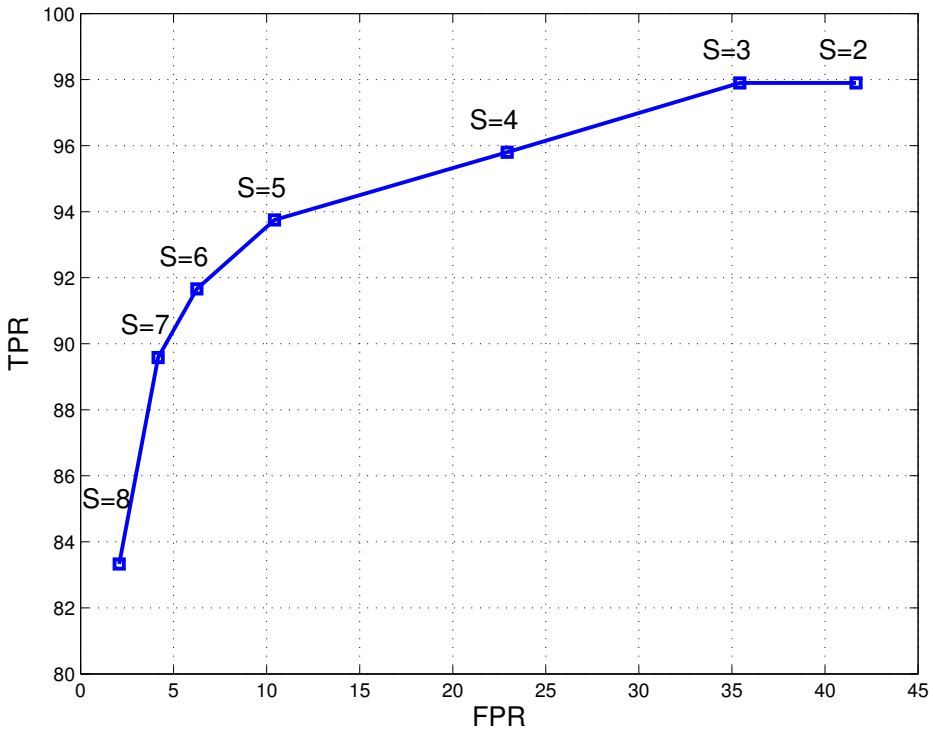


Figure 4: ROC curve: True positive vs. false positive rates for different sparsity parameter settings using a dictionary with 512 atoms.

the best tradeoff between the true positive and false positive rates can be achieved when feature size (S) equals 6.

3.3.4 Results on IMD Data Set

First, we evaluate the ability of our approach and the other approaches to detect plain CMF, i.e., a part of the image is copied and pasted in another part of the same image without any attack. An example of detection results on an image is shown in Fig.5. The detection results and average computation

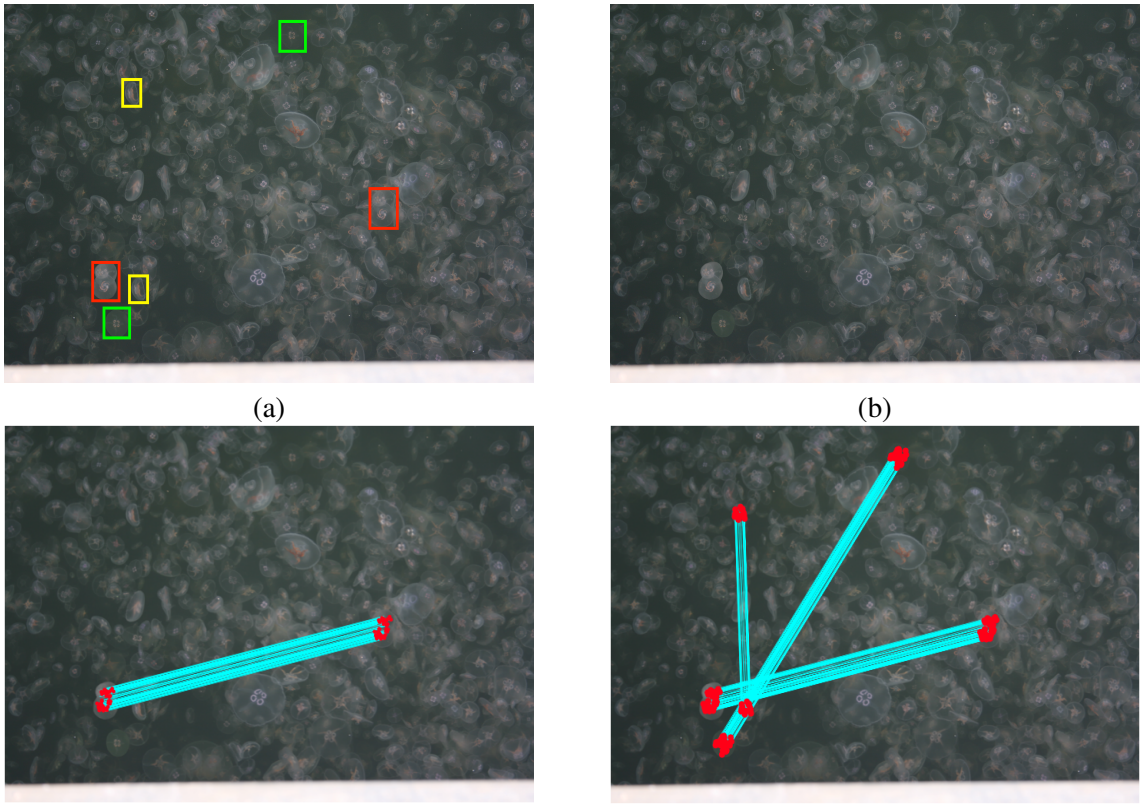


Figure 5: Detection results on an image from IMD data set. (a) Forged image with three tampered regions. (b) Detection result of G2NN-SIFT and PCA-SIFT. (c) Detection result of Segmented SIFT. (d) Detection result of the proposed approach.

times in seconds are shown in Table I. We observe that our approach not only outperforms the other approaches but also results in low computational time due to its low-dimensional feature vectors, i.e., 6.

Next, we evaluate the detection ability of our approach and the other approaches against three different attacks, including noise addition, JPEG compression, and rotation. The experimental results are

TABLE I: COMPARISON RESULTS OF PLAIN CMF DETECTION AND AVERAGE COMPUTATION TIMES PER IMAGE IN SECONDS FOR OUR PROPOSED APPROACH AND OTHER APPROACHES USING IMD DATA SET.

Method	Precision (%)	Recall (%)	$F1$	Time (s)
G2NN-SIFT [27]	88.4	79.2	83.5	610
PCA-SIFT [103]	81.8	75.0	78.3	214
Segmented SIFT [30]	70.2	83.3	76.2	719
Proposed	**93.6**	**91.7**	**92.6**	**146**

shown in Fig.6, which summarizes the detection results for different attacks. We observe that our approach drops linearly when large amounts of noise are added. However, our approach is more robust against JPEG compression and rotation, and it outperforms the other approaches.

TABLE II: COMPARISON RESULTS FOR OUR PROPOSED APPROACH AND OTHER APPROACHES USING MICC-F600 DATA SET.

Method	Precision (%)	Recall (%)	$F1$
G2NN-SIFT [27]	84.6	69.0	76.0
PCA-SIFT [103]	83.2	66.3	73.8
Segmented SIFT [30]	86.4	88.1	87.2
Proposed	**94.7**	**94.4**	**94.5**

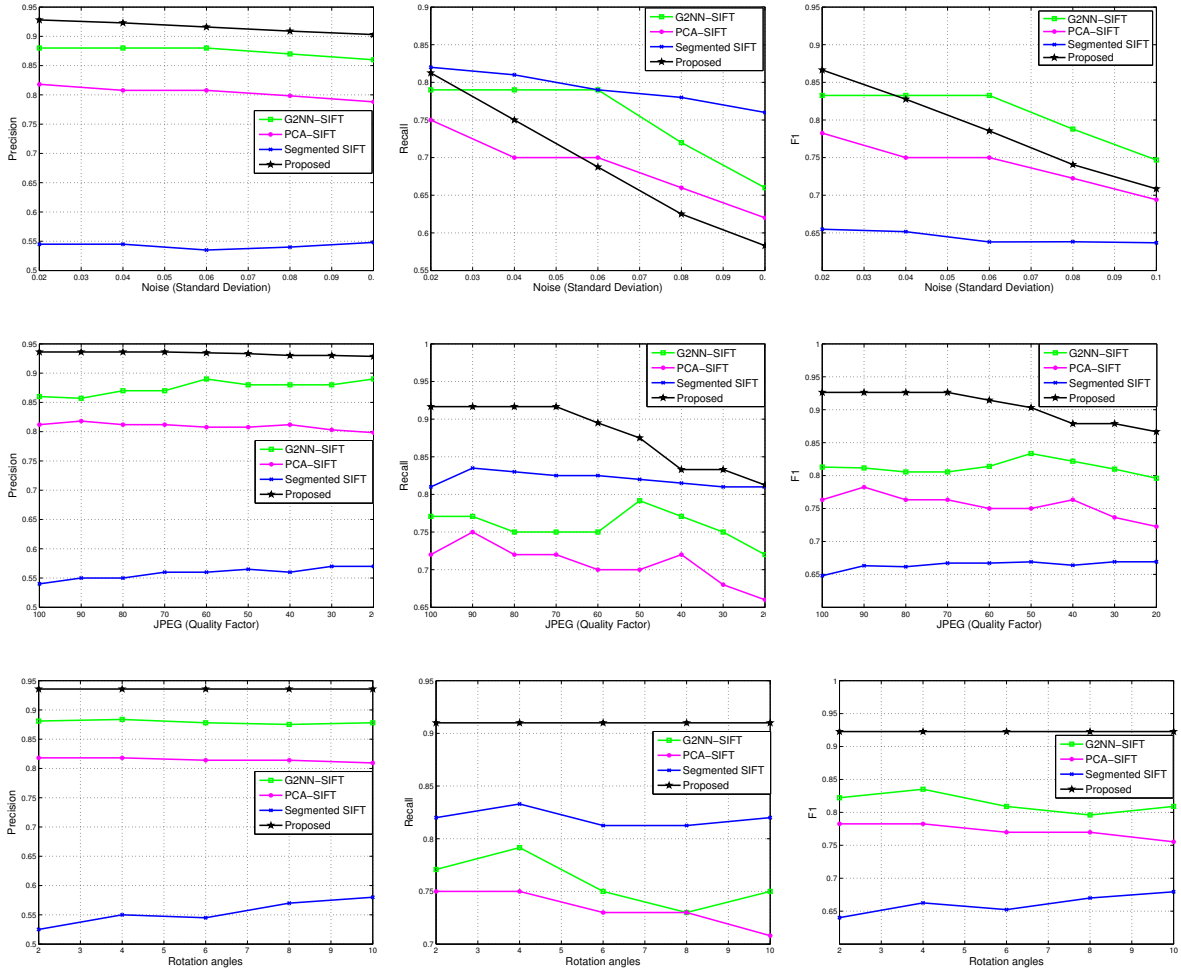


Figure 6: Performance comparison between our approach and the other approaches against three different attacks, which are adding noise, JPEG compression, and rotations. The three columns are corresponding to *Precision*, *Recall*, and $F1$ results, respectively. The three rows are corresponding to adding noise, JPEG compression, and rotations, respectively.

3.3.5 Results on MICC-F600 Data Set

We select this data set to evaluate the detection ability of our approach against combined attacks and large rotation angle, i.e., 30°. The detection results are shown in Table II in terms of *Precision*,

Recall, and $F1$, which suggests that our approach achieves superior performance compared to the other approaches. The average computation time is not reported in this table since the sizes of the images in the two data sets are the same.

CHAPTER 4

SEQUENTIAL AND PATCH ANALYSES FOR OBJECT REMOVAL VIDEO FORGERY DETECTION AND LOCALIZATION

In this chapter, we study the object-based video forgery problem, particularly removing objects from a video scene, as shown in Fig.7, and propose a novel approach based on sequential and patch analyses to detect object removal forgery and localize forged regions in videos. We perform sequential analysis by modeling video sequences as stochastic processes, where changes in the parameters of these processes indicate a video forgery. Patch analysis is performed by modeling video sequences as a mixture model of normal and anomalous patches, with the aim to separate these patches by identifying the distribution of each patch. Finally, we localize forged regions in videos by visualizing a movement of removed objects using anomalous patches.

4.1 Contributions

The contributions of this chapter can be summarized as follows:

1. We address a new and challenging object removal forgery problem when compared to frame-based forgery problem.
2. We model video sequences as multivariate processes to improve the detection accuracy.

Figure 7: An example of object removal video forgery: Images on the top row indicate frames from the original video; Images on the bottom row indicate corresponding frames from the tampered video where the man in the red box has been removed from the scene.

3. We model our patch analysis approach as a mixture model of normal and anomalous patches to further improve the detection accuracy.

4. We use the multivariate sequential and patch analyses to exponentially reduce the computational complexity. As a result, our approach is scalable.

5. We conduct our experiments at pixel and video levels. Hence, we further localize forged regions in videos.

The rest of this chapter is organized as follows: Our data model is explained in Sec. 4.2. Our proposed approach is described in Sec. 4.3. The spatiotemporal filter is presented in Sec. 4.3.1. Univariate and multivariate sequential analyses are presented in Sec. 4.3.2. Our patch analysis approach is explained in Sec. 4.3.3. Object removal visualization is described in Sec. 4.3.4. The experimental results are discussed in Sec. 4.4.

4.2 Data Model

We propose an approach based on sequential and patch analyses. Our approach requires the following assumptions about video sequences. First, video sequences are assumed to be captured by a static camera. Our approach aims to detect changes between video frames due to objects removal. When a camera is moving, it will generate video frames with different backgrounds. Thus, it would be hard to distinguish between changes due to movement of a camera or objects removal. Therefore, our approach mainly focuses on surveillance video clips where the camera is static. Second, video frames must be well-registered into a common reference frame prior to performing video forgery detection. We assume well-registered frames to eliminate changes due to non-registered frames.

In general, a pixel's intensity is corrupted by three sources of additive noise: photon counting noise, readout noise, and quantization noise [105, 106]. Photon counting noise comes from a discrete random number of photons striking the sensor and is modeled as a Poisson process. Readout noise is produced by the amplifier and is modeled as a Gaussian process. Quantization noise results from the selection of discrete pixel values and is modeled as a uniform distribution. It is extremely difficult to find the exact distribution of the additive noise that is added to pixel intensities [107]. Many previous works approximate this additive noise to be normally distributed [108]. Therefore, we assume pixels' values are drawn from a normal distribution, independent and identically distributed, and the variance remains constant throughout video frames while the mean is dependent on the scene.

The following notation and definitions will be used throughout this chapter. *Scalars* are written as lowercase letters, *vectors* are written as underlined lowercase letters, and *matrices* are written as uppercase letters. A *block* is defined as a group of spatially adjacent pixels, and it is described by a

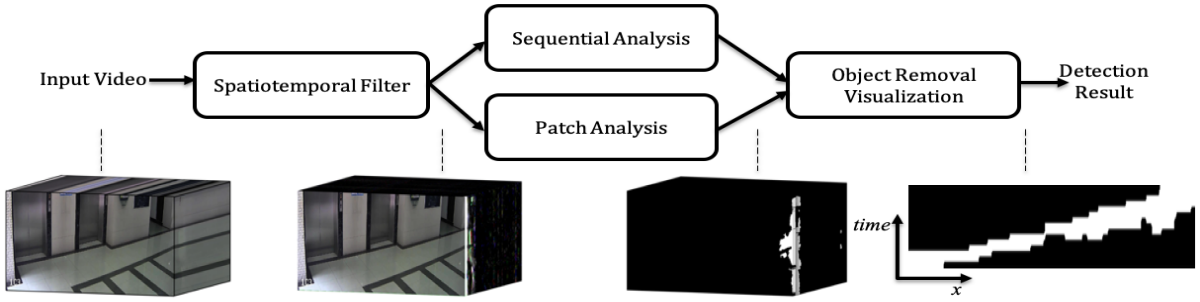


Figure 8: A flowchart of the proposed approach to object removal video forgery detection and localization.

feature vector. A *patch* is defined as a set of temporally adjacent blocks, and it is described by a set of feature vectors.

4.3 Object Removal Video Forgery Detection and Localization

We briefly describe our approach in the following steps, as illustrated in Fig.8. First, we apply spatial decomposition (i.e., Laplacian pyramid) to the video frames, followed by temporal high pass filter to detect edges spatially and highlight variations temporally. Then, we perform sequential analysis by modeling video sequences as stochastic processes, where changes in the parameters of these processes indicate a video forgery. If the patch analysis is performed, we model video sequences as a mixture model of normal and anomalous patches. These patches are subsequently separated by identifying if they have been generated by the normal or anomalous distribution. Finally, we localize forged regions by visualizing a movement of removed objects using anomalous patches.

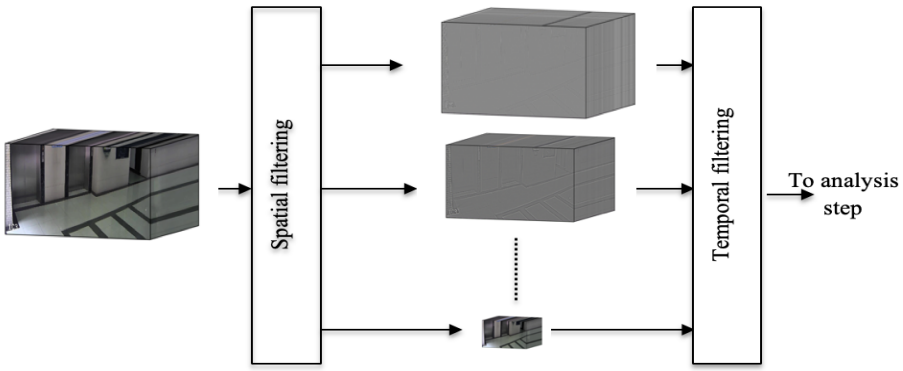


Figure 9: An overview of the proposed spatiotemporal filter.

4.3.1 Spatiotemporal Filter

We apply the spatiotemporal filter, which is presented in Fig.9, for two reasons. First, we use the spatiotemporal filter to expose traces (edges) that are left at a removed object boundary due to structure inpainting, texture inpainting, or a combination of the two. Second, we apply the spatiotemporal filter to zero out pixels' values at static regions, as shown in Fig.10. As a result, this filtering process enables sequential and patch analyses to accurately detect changes (i.e., anomalous) in forged videos.

Since the size of the removed objects is unknown, a video is divided into frames, and Laplacian pyramid decomposition [109] (spatial filtering) is applied to each frame to detect edges in all possible scales. The Laplacian pyramid decomposition subtracts each frame from its blurred version to form a video scale, down-samples each frame by half, and repeats this process until the minimum resolution of a frame is reached. This process constructs multiscale videos that represent edges at different scales, as shown in Fig.9. We perform temporal filtering at each scale by using the pixels' values throughout time in a frequency band and apply a high-pass filter to remove static edges.

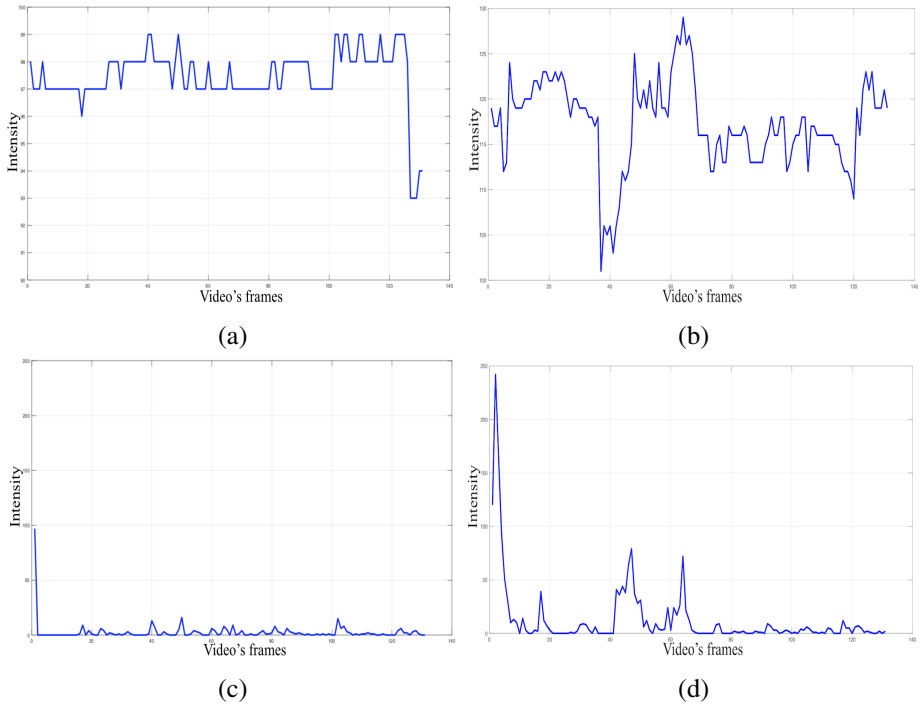


Figure 10: Intensity traces through video frames for (a) an authentic pixel before using the spatiotemporal filter (b) a forged pixel before using the spatiotemporal filter (c) an authentic pixel after using the spatiotemporal filter (d) a forged pixel after using the spatiotemporal filter.

4.3.2 Sequential Analysis

The spatiotemporal filter results in multiscale videos as shown in Fig.9, hence applying sequential analysis in each scale would result in very high computation time. Therefore, we first reconstruct the Laplacian pyramid to transfer multiscale videos to one video scale (i.e., the input video scale) [109]. The Laplacian pyramid reconstruction upsamples and blurs each frame in the lowest scale of Laplacian pyramid decomposition, adds the upsampled and blurred version to the next lowest scale to obtain the approximation of each frame at the next scale, and repeats this process until the input video scale is

reached. Then, we apply the following univariate or multivariate sequential analysis to the input video scale.

4.3.2.1 Univariate Analysis

We model the object removal forgery as an additive change in the mean value of probability density function associated with a pixel sequence in a video. We begin the analysis by introducing a null hypothesis H_0 that states there is no change in a pixel's mean value, and an alternative hypothesis H_1 that states there are changes in a pixel's mean value. The mean before the change μ_0 is assumed to be known, and the mean after the change μ_1 is assumed to be completely unknown but different than μ_0. We formulate the null and alternative hypotheses as follows:

$$\begin{aligned} \mathbf{H_0} &= \{\mu : \mu = \mu_0, n < t\} \\ \mathbf{H_1} &= \{\mu : \mu \neq \mu_0, n \geq t\} \end{aligned} \tag{4.1}$$

where n is the frame index, and t is the true change time. Based on our assumption that pixels' values are drawn from a normal distribution, independent and identically distributed as discussed in Sec. 4.2, we form the null and alternative likelihoods as follows:

$$\ell_k^{H_0}(x_i) = p(x_k, ..., x_n | H_0) = \frac{1}{\sqrt{2\pi\sigma^2}} \prod_{i=k}^{n} e^{\frac{-(x_i-\mu_0)^2}{2\sigma^2}} \tag{4.2}$$

$$\ell_k^{H_1}(x_i) = \sup_{\mu_1} p(x_k, ..., x_n | H_1) = \sup_{\mu_1} \frac{1}{\sqrt{2\pi\sigma^2}} \prod_{i=k}^{n} e^{\frac{-(x_i-\mu_1)^2}{2\sigma^2}} \tag{4.3}$$

where x_i represents values of a pixel throughout video frames; μ_i and σ^2 are the mean and variance of the pixel, respectively. Using (4.2) and (4.3), we form log-likelihood ratio as follows:

$$\Lambda_k^n = \ln \frac{\sup_{\mu_1} p(x_k, ..., x_n | H_1)}{p(x_k, ..., x_n | H_0)} \tag{4.4}$$

$$= \ln \frac{\sup_{\mu_1} \prod_{i=k}^{n} e^{\frac{-(x_i-\mu_1)^2}{2\sigma^2}}}{\prod_{i=k}^{n} e^{\frac{-(x_i-\mu_0)^2}{2\sigma^2}}} \tag{4.5}$$

The unknown mean is replaced by its maximum likelihood estimate (MLE) as follows:

$$\hat{x}_k^n = \frac{1}{n-k+1} \sum_{i=k}^{n} x_i. \tag{4.6}$$

Then, the log-likelihood ratio becomes

$$\Lambda_k^n = \frac{1}{2\sigma^2} \left[\sum_{i=k}^{n} (x_i - \mu_0)^2 - \sum_{i=k}^{n} (x_i - \hat{x}_k^n)^2 \right] \tag{4.7}$$

$$= \frac{1}{2\sigma^2} \sum_{i=k}^{n} \left[(x_i - \mu_0)^2 - (x_i - \hat{x}_k^n)^2 \right] \tag{4.8}$$

$$= \sum_{i=k}^{n} \left[\frac{(\hat{x}_k^n - \mu_0) x_i}{\sigma^2} + \frac{\mu_0^2 - \hat{x}_k^{n^2}}{2\sigma^2} \right]. \tag{4.9}$$

Then, the generalized log-likelihood g_k^n and alarm detection τ become

$$g_k^n = \max_{1 \le k \le n} \Lambda_k^n \tag{4.10}$$

$$\tau = \min\{n \geq 1 : g_k^n \geq h_u\}. \tag{4.11}$$

In (4.11), τ is a frame index where a change occurs, n is the discrete time index (frame index), and h_u is a threshold.

Let us summarize the univariate analysis. First, μ_0 and σ^2 are assumed to be known. In fact, they can be estimated using a pixel's values throughout all video frames. $\hat{x}_k^n$ is calculated sequentially by using all previous values of a pixel as described in (4.6). Finally, a change is declared if g_k^n exceeds a certain threshold h_u and this change is located at frame index τ.

4.3.2.2 Multivariate Analysis

We model the object removal forgery as an additive change in the mean parameter of probability density function associated with feature vectors that are extracted from dividing video frames into distinct blocks. We assume feature vectors are drawn from a Gaussian distribution, independent and identically distributed, with the following probability density function

$$p(\underline{y}_i) = \frac{1}{\sqrt{(2\pi)^r |\Sigma|}} e^{-\frac{1}{2}(\underline{y}_i - \underline{\mu})^T \Sigma^{-1} (\underline{y}_i - \underline{\mu})} \tag{4.12}$$

where $\underline{\mu}$ and Σ are the mean vector and covariance matrix of feature vectors, respectively; r is the dimension of the feature vector.

We begin with a general case [110] where the mean vector before the change $\underline{\mu_0}$ is limited by an upper bound, and mean vector after the change $\underline{\mu_1}$ is limited by a lower bound. Then, the null and alternative hypotheses become

$$\begin{aligned} \mathbf{H_0} &= \{\underline{\mu} : ||\underline{\mu} - \underline{\mu_0}||_\Sigma^2 \leq a^2, n < t\} \\ \mathbf{H_1} &= \{\underline{\mu} : ||\underline{\mu} - \underline{\mu_0}||_\Sigma^2 \geq b^2, n \geq t\} \end{aligned} \tag{4.13}$$

where $||\underline{\mu} - \underline{\mu_0}||_\Sigma^2 = \left(\underline{\mu} - \underline{\mu_0}\right)^T \Sigma^{-1} \left(\underline{\mu} - \underline{\mu_0}\right)$; t is the true change time; n is the frame index; $a < b$. Then, the log-likelihood ratio becomes

$$\Lambda_k^n = \ln \frac{\sup_{||\underline{\mu} - \underline{\mu_0}||_\Sigma \geq b} \prod_{i=k}^n p(\underline{y}_i)}{\sup_{||\underline{\mu} - \underline{\mu_0}||_\Sigma \leq a} \prod_{i=k}^n p(\underline{y}_i)} \tag{4.14}$$

$$= \ln \frac{\sup_{||\underline{\mu} - \underline{\mu_0}||_\Sigma \geq b} e^{-\frac{1}{2}\sum_{i=k}^n \left(\underline{y}_i - \underline{\mu}\right)^T \Sigma^{-1} \left(\underline{y}_i - \underline{\mu}\right)}}{\sup_{||\underline{\mu} - \underline{\mu_0}||_\Sigma \leq a} e^{-\frac{1}{2}\sum_{i=k}^n \left(\underline{y}_i - \underline{\mu}\right)^T \Sigma^{-1} \left(\underline{y}_i - \underline{\mu}\right)}} \tag{4.15}$$

$$\begin{aligned} &= \sup_{||\underline{\mu} - \underline{\mu_0}||_\Sigma \geq b} \left\{ -\frac{1}{2} \sum_{i=k}^n \left(\underline{y}_i - \underline{\mu}\right)^T \Sigma^{-1} \left(\underline{y}_i - \underline{\mu}\right) \right\} \\ &- \sup_{||\underline{\mu} - \underline{\mu_0}||_\Sigma \leq a} \left\{ -\frac{1}{2} \sum_{i=k}^n \left(\underline{y}_i - \underline{\mu}\right)^T \Sigma^{-1} \left(\underline{y}_i - \underline{\mu}\right) \right\}. \end{aligned} \tag{4.16}$$

The unknown parameter is replaced by its maximum likelihood estimate (MLE) as follows:

$$\underline{\hat{y}}_k^n = \frac{1}{n-k+1} \sum_{i=k}^n \underline{y}_i. \tag{4.17}$$

Then, the log-likelihood ratio becomes

$$\frac{2}{n-k+1}\Lambda_k^n = \begin{cases} -(Z_k^n - b)^2, & Z_k^n < a \\ -(Z_k^n - b)^2 + (Z_k^n - a)^2, & a \leq Z_k^n \leq b \\ (Z_k^n - a)^2, & Z_k^n > b \end{cases} \tag{4.18}$$

where Z_k^n is given by

$$Z_k^n = [\left(\underline{\hat{y}}_k^n - \underline{\mu_0}\right)^T \Sigma^{-1} \left(\underline{\hat{y}}_k^n - \underline{\mu_0}\right)]^{1/2}. \tag{4.19}$$

We set $a = b = 0$ in (4.18) because we are interested in the case where the mean vector before the change $\underline{\mu_0}$ is assumed to be known and the mean vector after the change $\underline{\mu_1}$ is assumed to be completely unknown but different than $\underline{\mu_0}$. Then, the generalized log-likelihood g_k^n and alarm detection τ become

$$g_k^n = \max_{1 \leq k \leq n} \{\frac{n-k+1}{2}(Z_k^n)^2\} \tag{4.20}$$

$$\tau = \min\{n \geq 1 : g_k^n \geq h_m\}. \tag{4.21}$$

Let us summarize the multivariate analysis. First, $\underline{\mu_0}$ and Σ are assumed to be known. In fact, they can be estimated using feature vectors of a particular block throughout all video frames. $\underline{\hat{y}}_k^n$ and Z_k^n are calculated sequentially by using all previous feature vectors of a particular block as described in

(4.17) and (4.19), respectively. Finally, a change is declared if g_k^n exceeds a certain threshold h_m and this change is located at frame index τ.

The current formulation of univariate and multivariate analyses enables us to detect only a single change in the whole time (frame) series. However, we need to detect multiple changes, hence we use binary segmentation [111]. Binary segmentation starts by detecting a single change in the complete time series. If there is a change, it splits the time series around this change into two sub-series and repeats this process until no changes are detected. By using binary segmentation, the time series that represents video frames will be divided into segments.

A segment is considered as a forged segment (removed object segment) if two conditions are met: (1) the mean of this segment exceeds a certain threshold to identify whether this segment belongs to a background or a removed object, and (2) the length of this segment is less than a certain threshold based on our definition that removed objects are moving as discussed in Chapter 1.

4.3.3 Patch Analysis

We model the object removal forgery as a mixture model of normal and anomalous patches. A *patch* is defined as a set of temporally adjacent blocks, and it is described by a set of feature vectors. We assume that all feature vectors in a patch are either normal or anomalous. Some patches that are located at the border between forged and original regions may contain feature vectors that belong to both normal and anomalous sets. However, these patches are few because forged regions are generally small compared to the original regions in a video. Hence, we neglect patches at the border and consider only patches that contain either normal or anomalous feature vectors. We also assume that normal features

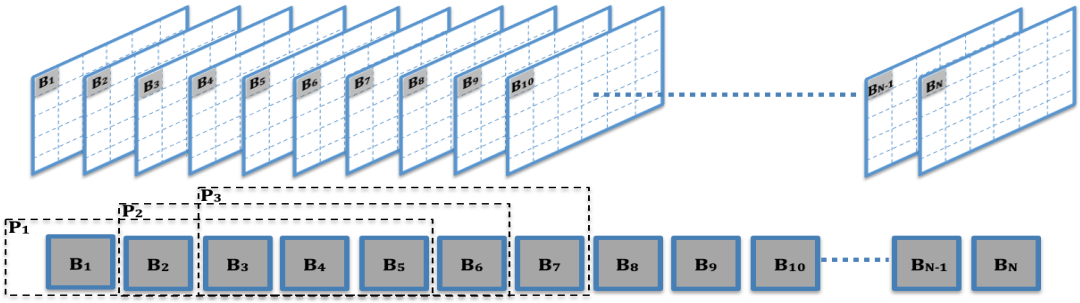


Figure 11: Illustration of the proposed patch analysis approach. Top sequence shows video frames that are divided into non overlapping blocks with a selected gray block to apply patch analysis; bottom sequence indicates patch size $p = 5$ with overlapping step $s = 1$. Patch analysis starts by calculating the log-likelihood under the assumption that all blocks belong to normal set. Then, it calculates the log-likelihood under the assumption that P_1 belongs to anomalous set. If the difference between these two log-likelihoods is less than a threshold h_p, P_1 is moved from normal set to anomalous set. Otherwise, P_1 remains in the normal set. The analysis is repeated for P_2, P_3,...,P_{N-p}.

are drawn from a Gaussian distribution, and anomalous features are drawn from a uniform distribution because anomalies are often assumed to be uniform [112, 113]. The probability density functions for normal $p_N(\underline{y})$ and anomalous $p_A(\underline{y})$ feature vectors are defined as

$$p_N(\underline{y}_i) = \frac{1}{\sqrt{(2\pi)^r |\Sigma|}} e^{-\frac{1}{2}(\underline{y}_i - \underline{\mu})^T \Sigma^{-1} (\underline{y}_i - \underline{\mu})} \tag{4.22}$$

$$p_A(\underline{y}_i) = \begin{cases} \frac{1}{(b-a)^r}, \underline{y}_i \in (a,b)^r \\ 0, Otherwise \end{cases} \tag{4.23}$$

where $\underline{\mu}$ and Σ are the mean vector and covariance matrix of feature vectors, respectively; r is the dimension of the feature vector; a and b are the minimum and maximum values of arbitrary feature vectors, respectively. We let N_k and A_k be the sets of normal and anomalous feature vectors, at frame

index k, respectively. Initially, all feature vectors of a particular block are put in a normal set while an anomalous set is empty.

We begin with a general case where the null hypothesis H_0 states that there is at least one feature vector $\underline{y}_i$ in a patch belongs to a normal set and an alternative hypothesis H_1 states that all feature vectors $\underline{y}_i$ in the patch belong to an anomalous set. We formulate the null and alternative hypotheses as follows:

$$\begin{aligned} \mathbf{H_0} &= \{\exists \underline{y}_i \in C, \underline{y}_i \sim p_N(\underline{y}_i)\} \\ \mathbf{H_1} &= \{\forall \underline{y}_i \in C, \underline{y}_i \sim p_A(\underline{y}_i)\} \end{aligned} \tag{4.24}$$

where $C = \{y_{k+1}, ..., y_{k+p}\}$ is a patch that consists of p feature vectors. By assuming that the patches are generated in an independent manner, the likelihood of null ($\ell_k^{H_0}(\underline{y})$) and alternative ($\ell_k^{H_1}(\underline{y})$) hypotheses of the entire feature vectors for a particular block at an arbitrary frame k are as follows:

$$\ell_k^{H_0}(\underline{y}) = \sum_{c_i \in (\mathbf{P}(C) - \{C\})} (1-\lambda)^{|N_{k-1} - c_i|} \prod_{\underline{y}_i \in (N_{k-1} - c_i)} p_N(\underline{y}_i) \tag{4.25}$$

$$\left((\lambda)^{|A_{k-1} \cup c_i|} \prod_{\underline{y}_i \in (A_{k-1} \cup c_i)} p_A(\underline{y}_i) \right)$$

$$= (1-\lambda)^{|N_{k-1}|} \prod_{\underline{y}_i \in N_{k-1}} p_N(\underline{y}_i) (\lambda)^{|A_{k-1}|} \tag{4.26}$$

$$\prod_{\underline{y}_i \in A_{k-1}} p_A(\underline{y}_i) \sum_{c_i \in (\mathbf{P}(C) - \{C\})} \left(\frac{\lambda}{1-\lambda}\right)^{|c_i|} \prod_{\underline{y}_j \in c_i} \frac{p_A(\underline{y}_j)}{p_N(\underline{y}_j)}$$

(4.27)

$$\ell_k^{H_1}(\underline{y}) = (1-\lambda)^{|N_{k-1}-C|} \prod_{\underline{y}_i \in (N_{k-1}-C)} p_N(\underline{y}_i) \tag{4.28}$$

$$\Big((\lambda)^{|A_{k-1}\cup C|} \prod_{\underline{y}_i \in (A_{k-1}\cup C)} p_A(\underline{y}_i)\Big)$$

$$= (1-\lambda)^{|N_{k-1}|} \prod_{\underline{y}_i \in N_{k-1}} p_N(\underline{y}_i)(\lambda)^{|A_{k-1}|} \tag{4.29}$$

$$\prod_{\underline{y}_i \in A_{k-1}} p_A(\underline{y}_i)\Big((\frac{\lambda}{1-\lambda})^{|C|} \prod_{\underline{y}_i \in C} \frac{p_A(\underline{y}_i)}{p_N(\underline{y}_i)}\Big)$$

where $\mathbf{P}(C)$ is the power set of C, which is the set of all subsets of C; $|\cdot|$ is the cardinality of a set; and λ is the expected fraction of anomalies. Then, the likelihood ratio becomes

$$\Lambda_k = \frac{\ell_k^{H_0}(\underline{y})}{\ell_k^{H_1}(\underline{y})} = (\frac{1-\lambda}{\lambda})^{|C|} \prod_{\underline{y}_i \in C} \frac{p_N(\underline{y}_i)}{p_A(\underline{y}_i)} \tag{4.30}$$

$$+ \sum_{c_i \in (\mathbf{P}(C)-\{C,\{\}\})} (\frac{1-\lambda}{\lambda})^{|c_i|} \prod_{\underline{y}_j \in c_i} \frac{p_N(\underline{y}_j)}{p_A(\underline{y}_j)}.$$

Based on our assumption that all feature vectors in a patch are either normal or anomalous, the probability of the second term in (4.30) to occur is zero. Hence, the likelihood and log-likelihood become

$$\Lambda_k = (\frac{1-\lambda}{\lambda})^{|C|} \prod_{\underline{y}_i \in C} \frac{p_N(\underline{y}_i)}{p_A(\underline{y}_i)} \tag{4.31}$$

$$\ln(\Lambda_k) = |C| \ln(\frac{1-\lambda}{\lambda}) + \sum_{\underline{y}_i \in C} p_N(\underline{y}_i) - \sum_{\underline{y}_i \in C} p_A(\underline{y}_i) \underset{H_1}{\overset{H_0}{\gtrless}} h_p \tag{4.32}$$

where h_p is the decision threshold.

Let us summarize patch analysis as described in Algorithm 1. Initially, all feature vectors of a particular block are put in a normal set while an anomalous set is empty. A patch is chosen in an overlapping manner with overlapping step $s = 1$. Then, we calculate the likelihood and log-likelihood using equations described in (4.31) and (4.32), respectively. If the log-likelihood is less than a threshold h_p, this patch is declared as an anomaly and it is moved from the normal set to the anomalous set. Otherwise, this patch remains in the normal set. The main steps of patch analysis approach are illustrated in Fig.11.

4.3.4 Object Removal Visualization

We construct a binary video where a pixel equals one in frames that belong to anomalous sets (changed segments) and equals zero in frames that belong to normal sets (unchanged segments). A video forgery is detected if a number of consecutive frames h_F have an area that is larger than a threshold h_A and contains only ones. In the experiment, we set h_F to 25 frames and h_A to 500 pixels based on the results of ROC curve shown in Fig.12b.

We localize the movement of removed objects by constructing another binary video where a pixel equals one in a frame where a change occurs until the last video frame and equals zero in the other frames; essentially, once a pixel's value becomes one, it maintains that value until the last video frame. This process will create paths of removed objects; these paths can be visualized by plotting the last spatiotemporal XT slice (width vs. time), which is a bird's-eye view of a video as shown in Fig.8.

Algorithm 1 Object Removal Forgery Detection and Localization Based on Patch Analysis

Require: V ▷ Input video
b ▷ Block size
p ▷ Patch size
Ensure: M ▷ Object removal visualization result
$N_f \leftarrow NumberOfFrames$
$R \leftarrow EmptyArray$ ▷ Size of R is the same as size of V
$Apply\ spatiotemporal\ filter\ on\ V$
$Divide\ R\ and\ filtered\ V\ into\ distinct\ blocks\ b \times b$
for *each block* (B_{Vi}) *in* V **do**
    $\{\underline{y}_1, ..., \underline{y}_{N_f}\} \leftarrow GetAllFeatureVectorsThrough\ V$
    $Initialization: k = 0, N_0 = \{\underline{y}_1, ..., \underline{y}_{N_f}\}, A_0 = \{\}$
    while $k \leq N_f - p$ **do**
        $C = \{y_{k+1}, ..., y_{k+p}\}$
        $Compute\ likelihood\ ratio\ \Lambda_k\ based\ on\ (4.31)$
        $Compute\ log-likelihood\ \ln(\Lambda_k)\ based\ on\ (4.32)$
        if $\ln(\Lambda_k) < h_p$ **then**
            $N_k = N_{k-1} - \{y_{k+1}, ..., y_{k+p}\}$
            $A_k = A_{k-1} \cup \{y_{k+1}, ..., y_{k+p}\}$
            $k = k + p$ ▷ C is an anomalous patch, hence k is increased by p
        else
            $N_k = N_{k-1}$
            $A_k = A_{k-1}$
            $k = k + 1$ ▷ C is a normal patch, hence k is increased by 1
        end if
    end while
    $B_{Ri} \leftarrow BinaryArray$ ▷ Explained in Sec.4.3.4
end for ▷ $B_{Ri} \equiv$ Corresponding block in R
$M \leftarrow RemovalVisualization$ ▷ Explained in Sec.4.3.4

4.4 Experimental analysis

In this section, we describe the data set and detection performance measurements. We also analyze the results of our approach and compare our approach with state-of-the-art approaches. We carry out our experiments using a MacBook Pro with 2.9 GHz Intel dual core i7 CPU and 8 GB RAM.

4.4.1 Data Set

To the best of our knowledge, the only available video forgery data sets are SULFA [114] and SYSU-OBJFORG [70]. SULFA is a frame-based forgery, which is beyond the scope of this work. Therefore, we use SYSU-OBJFORG, where all videos are extracted from a static surveillance camera with a resolution of 1280×720 and 25 frames per second. This data set consists of 100 original videos and 100 object-based forged videos; each video is approximately 11 seconds in duration. According to the authors of [70], SYSU-OBJFORG is the largest object-based forged video data set in the literature. However, most of the forged videos are not realistic because the counterfeit regions can be identified using the naked eye. In other words, object-based forgery is performed in the middle of frames, e.g., a walking person is removed before leaving a video scene, so this person is seen for a couple of seconds and suddenly disappeared from the video scene. Hence, we use SYSU-OBJFORG data set to generate realistic object removal forged videos by using two recent inpainting algorithms [61, 63]. Fig.13 shows three examples of object removal forgery from the data set.

To evaluate the effectiveness of our approach, we generate three video sets from the data set. The first set is an uncompressed video set, which has object removal forged videos without compressing these videos. The second set is a compressed video set, which has object removal forged videos with

compressing these videos using H.264/MPEG-4 with 1 Mbps. The third set is a low-resolution video set, which has object removal forged videos with reducing the original resolution by half, i.e., 640×360.

4.4.2 Evaluation Metric

We evaluate object removal forgery detection on video and pixel levels. The most important aspect in practice is to determine whether a video is forged or not, i.e., video level performance. However, the effectiveness of an algorithm is determined by how accurately the tampered regions can be identified in a video, i.e., pixel level performance. We measure the performance at video level by defining T_P as the correctly detected forged videos, F_P as original videos that have been incorrectly detected as forged, and F_N as falsely missed forged videos. Then, *Precision*, *Recall*, $F1$, and *Intersection over Union* (IoU) are as follows:

$$Precision = \frac{T_P}{T_P + F_P} \tag{4.33}$$

$$Recall = \frac{T_P}{T_P + F_N} \tag{4.34}$$

$$F1 = \frac{2T_P}{2T_P + F_P + F_N} \tag{4.35}$$

$$IoU = \frac{T_P}{T_P + F_P + F_N} \tag{4.36}$$

Precision shows the probability that a detected forgery is truly a forgery, *Recall* indicates the probability that a forged video is detected, $F1$ score shows the average performance, and IoU score shows the worst case performance.

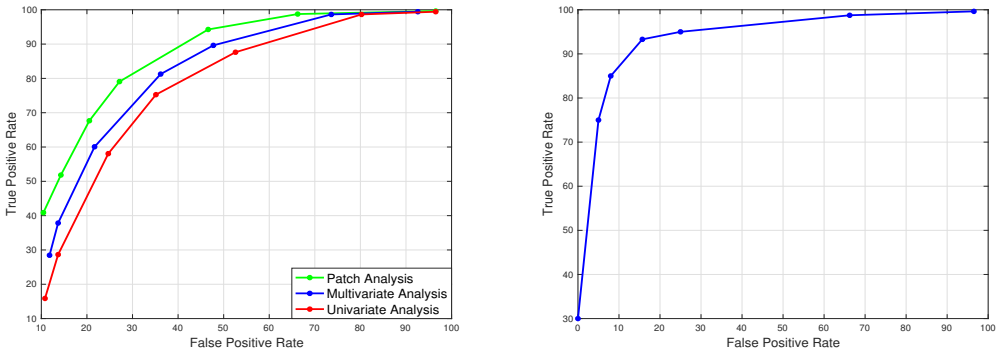
We measure the performance at pixel level by defining T_P as the correctly detected forged pixels, F_P as original pixels that have been incorrectly detected as forged, and F_N as falsely missed forged pixels. Then, we compute *Precision*, *Recall*, $F1$, and IoU as in (4.33), (4.34), (4.35), and (4.36) respectively.

4.4.3 Feature Selection

There are several feature extraction approaches that have been used to detect image forgery such as SIFT [115], and SURF [116]. However, these approaches lead to a high dimensional feature vector that reduces detectability of changes, especially if this vector contains irrelevant features [117]. One way to overcome this problem is to use one of dimension reduction approaches such as PCA [118], but these approaches often result in loss of relevant features. Since feature vectors are required to be relevant with small size, we experimentally observe that *mean* and *variance* are relevant features to our model. In particular, we observe that removed object traces that are exposed by the proposed spatiotemporal filter disrupt *mean* and *variance* of video frame blocks. Therefore, we believe that *mean* and *variance* are appropriate features for our model. As a result, we compute the *mean* and *variance* for each block in video frames and use them as feature vectors throughout multivariate and patch analyses.

4.4.4 Threshold Settings

Initially, we choose 10% of the dataset to tune threshold values of univariate (h_u), multivariate (h_m), and patch (h_p) analyses. The receiver operating characteristic (ROC) curves that are illustrated in Fig.12a suggest the best tradeoff between the true positive and false positive rates at pixel level for the three analyses can be achieved when $h_u = 15$, $h_m = 35$, and $h_p = 10$. Thus, we choose these threshold values based on the results of ROC curves.



(a) (b)

Figure 12: ROC curves: True positive vs. false positive rates at (a) pixel level for different change thresholds using univariate, multivariate, and patch analyses, and (b) at video level using different thresholds for the number of consecutive frames (h_F) and areas (h_A).

We also tune threshold values for the number of consecutive frames (h_F) and areas (h_A) that are used to declare forgery at video level. We start with $h_F = 5$ with an increment of 10 frames and $h_A = 300$ with an increment of 100 pixels. The ROC curve that is shown in Fig.12b suggests the best tradeoff between the true positive and false positive rates can be achieved when $h_F = 25$ and $h_A = 500$. Thus, we selected these threshold values for all experiments.

4.4.5 Detection Results

We evaluate detection results at both pixel and video levels using sequential analysis, followed by patch analysis.

4.4.5.1 Results of Sequential Analysis

We evaluate both the effectiveness of (1) the univariate analysis when the block size equals one, and (2) the multivariate analysis with changes in block size.

Detection results of the uncompressed video set at pixel and video levels for different block sizes are shown in Table IIIa. We observe that Recall and F1 values at the pixel level increase when the block size increases until they reach their largest values at the block size equals 10. Then, these values slightly decrease, which suggests that the optimal block size is 10. Similarly, Recall and F1 values at the video level follow the same pattern with better detection results because detecting only one forged second (i.e., 25 frames) is enough to declare that a video is forged as discussed in Sec.4.4.4.

Detection results of the compressed video set at pixel and video levels for different block sizes are shown in Table IIIb. We observe that the largest Precision value at the pixel level occurs at the block size equals one, which indicates that the false positive rate increases when the block size increases. The largest F1 and IoU scores at both pixel and video levels happen at the block size equals 15, which suggests that the optimal block size is 15. Furthermore, we notice that detection results are still high even though videos in this video set are compressed, which indicates that the sequential analysis is robust against compressed videos.

Detection results of the low-resolution video set at pixel and video levels for different block sizes are shown in Table IIIc. We observe that the smallest Precision value at both pixel and video levels occurs at the block size equals five. However, the largest Recall and F1 values at both pixel and video levels happen at the block size equals five, which suggests that the optimal block size is five. Moreover, we notice that detection results are still high even though videos in this video set have low resolutions, which indicates that the sequential analysis is also robust against lower resolution videos.

We observe that detection results at video level are better than detection results at pixel level. This result is expected because it is enough to detect a small number (i.e., 25) of consecutive forged frames to

TABLE III: DETECTION RESULTS AT PIXEL AND VIDEO LEVELS OF OBJECT REMOVAL VIDEO FORGERY USING SEQUENTIAL ANALYSIS WITH DIFFERENT BLOCK SIZES AND DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF UNCOMPRESSED VIDEO SET.

Block size	Pixel level				Video level			
	Precision (%)	Recall (%)	F1 (%)	IoU (%)	Precision (%)	Recall (%)	F1 (%)	IoU (%)
1	**76.54**	78.79	77.65	63.47	93.33	93.33	93.33	87.49
5	73.68	82.48	77.84	63.72	91.40	94.44	92.90	86.73
10	73.62	**85.80**	**79.25**	**65.63**	93.48	**95.56**	**94.51**	**89.58**
15	73.01	85.26	78.66	64.83	92.22	92.22	92.22	85.57
20	74.51	82.45	78.28	64.31	**94.25**	91.11	92.66	86.32

(b) DETECTION RESULTS OF COMPRESSED VIDEO SET.

Block size	Pixel level				Video level			
	Precision (%)	Recall (%)	F1 (%)	IoU (%)	Precision (%)	Recall (%)	F1 (%)	IoU (%)
1	**75.78**	78.50	77.11	62.75	92.22	92.22	92.22	85.57
5	74.26	80.69	77.34	63.05	89.36	93.33	91.30	84.00
10	73.29	**84.15**	78.34	64.39	91.30	93.33	92.31	85.71
15	74.00	83.23	**78.35**	**64.41**	92.39	**94.44**	**93.41**	**87.63**
20	75.31	81.03	78.07	64.03	**93.18**	91.11	92.13	85.42

(c) DETECTION RESULTS OF LOW-RESOLUTION VIDEO SET.

Block size	Pixel level				Video level			
	Precision (%)	Recall (%)	F1 (%)	IoU (%)	Precision (%)	Recall (%)	F1 (%)	IoU (%)
1	**76.06**	75.89	75.97	61.25	**92.86**	86.67	89.66	81.25
5	71.30	**84.77**	**77.45**	**63.20**	89.36	**93.33**	**91.30**	**84.00**
10	73.02	82.07	77.28	62.97	90.11	91.11	90.61	82.83
15	75.61	75.63	75.62	60.80	91.86	87.78	89.77	81.44
20	75.16	72.82	73.98	58.70	92.77	85.56	89.02	80.21

declare forgery at video level (i.e., video forgery detection) as discussed in Sec.4.4.4, whereas detecting forgery at pixel level requires to detect all forged pixels, which is often significantly large, to localize forged regions in a video (i.e., video forgery localization).

In summary, we observe that the optimal block sizes of uncompressed, compressed, and low-resolution video sets are ten, fifteen, and five, respectively, because these block sizes lead to the highest detection performance (F1 score). We also consider ten as the optimal block size of compressed video set because F1 scores are almost the same at block sizes equal ten and fifteen. Therefore, the optimal block size of uncompressed video set is the same as the optimal block size of compressed video set because these video sets have the same resolution (i.e., 1280×720). The optimal block size of low-resolution video set is half of the optimal block size of uncompressed video set, which is expected because the resolution of low-resolution video set is reduced by half compared to the resolution of uncompressed video set. The highest detection performance is achieved when using the uncompressed video set, and then it slightly decreases throughout the compressed and low-resolution video sets. The overall detection performance (F1 score) of the three video sets is improved when using the multivariate analysis compared to the univariate analysis, which is one of our key contributions. We believe the reason for this improvement is that forged regions in a video are always larger than one pixel, hence applying multivariate analysis, which is based on blocks, increases detection results throughout the three video sets.

4.4.5.2 Results of Patch Analysis

We need to fix the block size while the patch size is varied in order to evaluate the effectiveness of the patch analysis. We notice that the optimal block size is not the same for the three video sets. However, the difference between the largest F1 score and the other F1 scores at the block size equals 10 is very small across the three video sets. Hence, we set the block size to 10 throughout the patch analysis.

Detection results of the uncompressed video set at pixel and video levels for different patch sizes are shown in Table IVa. We observe that the Recall value at the pixel level peaks when the patch size is eight, and then subsequently decreases, indicating that the false negative rate increases as the patch size increases. The largest F1 and IoU scores at both pixel and video levels happen at the patch size equals 12, which suggests that the optimal patch size is 12. We also notice that when the detection results at the pixel level improve, the detection results at the video level improve as well, which is expected because the pixel is a fundamental unit of videos.

Detection results of the compressed video set at pixel and video levels for different patch sizes are shown in Table IVb. We observe that the largest Recall value at the pixel level occurs at the patch size equals four, which indicates that the true positive rate does not improve when the patch size increases. The largest Precision value at both pixel and video levels occurs at the same patch size, which is 12. However, the largest F1 score at pixel and video levels happens at the patch sizes equal 12 and eight, respectively. Hence, the optimal patch size at pixel level is not the same as the optimal patch size at video level.

Detection results of the low-resolution video set at pixel and video levels for different patch sizes are shown in Table IVc. We observe that Precision value at both pixel and video level increases when the patch size increases until it reaches its largest value at the patch size equals eight. Then, its value slightly decreases, which indicates that the false positive rate increases when the patch size increases beyond eight. The largest F1 score at pixel and video levels happens at the patch sizes equal eight and twelve, respectively. However, at the pixel level, the difference between F1 score at patch size equals eight and twelve is very small. Hence, the optimal patch size at both pixel and video levels is 12.

TABLE IV: DETECTION RESULTS AT PIXEL AND VIDEO LEVELS OF OBJECT REMOVAL VIDEO FORGERY USING PATCH ANALYSIS WITH DIFFERENT PATCH SIZES AND DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF UNCOMPRESSED VIDEO SET.

Patch size	Pixel level				Video level			
	Precision (%)	Recall (%)	F1 (%)	IoU (%)	Precision (%)	Recall (%)	F1 (%)	IoU (%)
4	70.32	88.82	78.49	64.60	90.63	96.67	93.55	87.88
8	74.17	**89.19**	80.99	68.05	92.55	96.67	94.57	89.69
12	**74.49**	88.90	**81.06**	**68.15**	**95.65**	**97.78**	**96.70**	**93.62**
16	73.52	86.94	79.67	66.21	93.41	94.44	93.92	88.54
20	72.57	85.38	78.45	64.54	94.32	92.22	93.26	87.37

(b) DETECTION RESULTS OF COMPRESSED VIDEO SET.

Patch size	Pixel level				Video level			
	Precision (%)	Recall (%)	F1 (%)	IoU (%)	Precision (%)	Recall (%)	F1 (%)	IoU (%)
4	71.31	**87.91**	78.74	64.93	89.58	95.56	92.47	86.00
8	74.80	87.35	80.59	67.49	93.55	**96.67**	**95.08**	**90.63**
12	**75.17**	87.08	**80.69**	**67.63**	**95.51**	94.44	94.97	90.43
16	74.53	85.18	79.50	65.98	94.38	93.33	93.85	88.42
20	73.36	82.58	77.70	63.53	90.11	91.11	90.61	82.83

(c) DETECTION RESULTS OF LOW-RESOLUTION VIDEO SET.

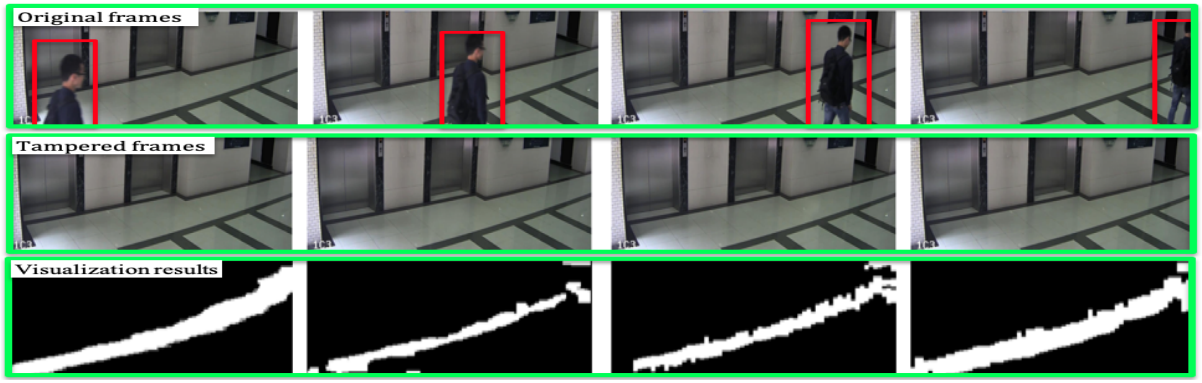
Patch size	Pixel level				Video level			
	Precision (%)	Recall (%)	F1 (%)	IoU (%)	Precision (%)	Recall (%)	F1 (%)	IoU (%)
4	78.94	78.88	78.91	65.17	92.05	90.00	91.01	83.51
8	**79.38**	81.57	**80.46**	**67.31**	**94.38**	93.33	93.85	88.42
12	78.58	**82.38**	80.44	67.28	93.48	**95.56**	**94.51**	**89.58**
16	78.19	79.77	78.97	65.25	91.40	94.44	92.90	86.73
20	77.67	78.94	78.30	64.34	91.21	92.22	91.71	84.69

As can be seen from Table IV, the pixel level performance of our patch analysis approach (which is currently one of few approaches available for detection of object removal video forgery) achieves a detection rate of only 81.06%, therefore the data set employed provides a challenging framework for evaluation of the detection of object removal video forgery.

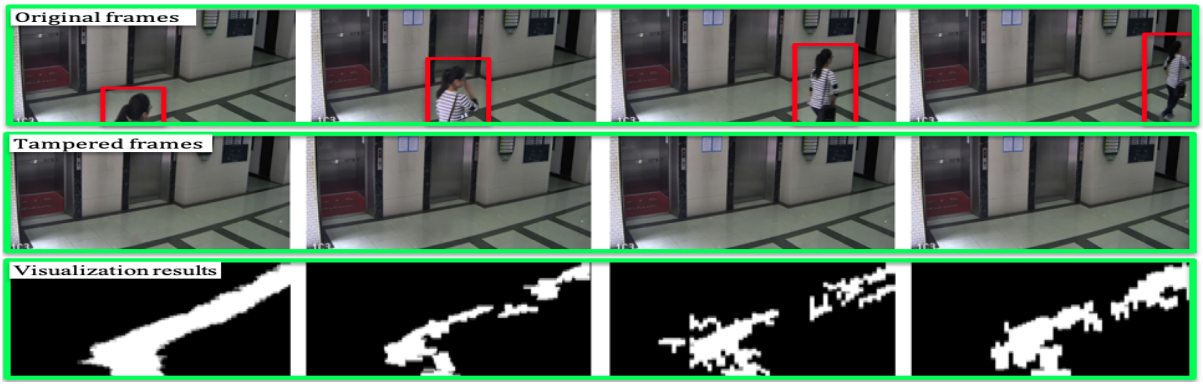
In summary, we consider that the optimal patch size of uncompressed, compressed, and low-resolution video sets is twelve because this patch size leads to the highest detection performance (F1 score). We observe that there are no significant differences between the largest F1 scores in the three video sets. Hence, our patch analysis approach is robust against compressed and lower resolution videos. We believe that there are two reasons for this robustness. First, the proposed spatiotemporal filter is able to expose traces that are left at a removed object boundary even though videos are compressed or have low resolutions. Second, compression or low resolution is applied to all video frames, hence all patches are compressed or have low resolutions. As a result, our patch analysis can distinguish between normal and anomalous patches in a video if the attack (e.g., compression) is applied to all patches. The highest detection performance is achieved when using the uncompressed video set, and then it slightly decreases throughout the compressed and low-resolution video sets. The overall detection performance (F1 score) of the three video sets is further improved when using our patch analysis approach, which is another major contribution of this work. We believe the reason for this improvement is that object removal forgery always happens in temporally adjacent regions, hence applying patch analysis, which is based on temporally adjacent blocks, increases detection results throughout the three video sets.

4.4.5.3 Comparison Between Sequential and Patch Analyses

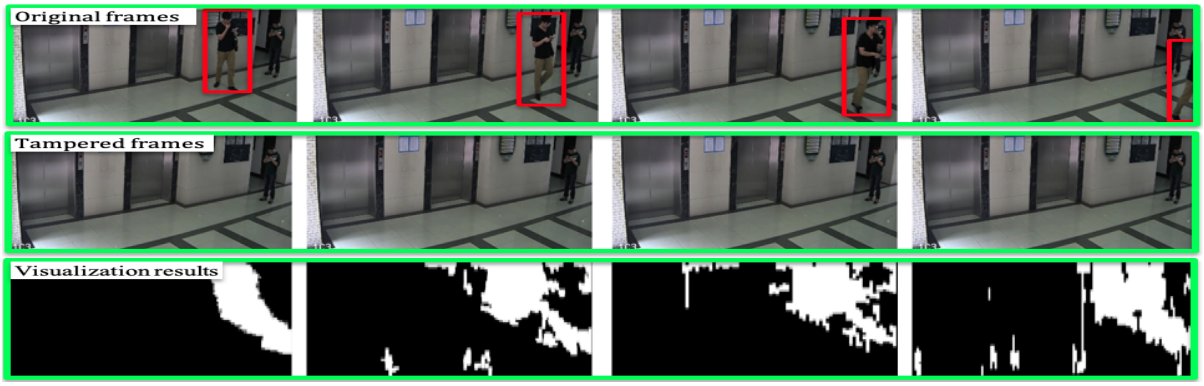
We set the block size to 10 and the patch size to 12 in this comparison because these values are optimal for block and patch sizes based on the results in Table III and Table IV.



(a)



(b)



(c)

Figure 13: Three examples of visualization results of a removed object movement using the univariate, multivariate, and patch analyses. In each example, images on the top row indicate frames from the original video; images on the middle row indicate the corresponding frames from the tampered video where the object in the red box has been removed from the scene; images on the bottom row (from left to right) indicate the ground truth of the removed object movement, the movement visualization using the univariate analysis, multivariate analysis, and patch analysis, respectively.

We compare forgery detection between sequential and patch analyses by plotting receiver operating characteristic (ROC) curves. Fig.12a shows ROC curves for different change thresholds (h_u, h_m, h_p) using univariate, multivariate, and patch analyses. We observe that our patch analysis outperforms univariate and multivariate analyses throughout different change thresholds, which is expected because detection results in Table IV are better than detection results in Table III. We believe that our patch analysis outperforms sequential analysis because our patch analysis is based on spatiotemporal analysis, whereas sequential analysis is based on spatial analysis. Hence, our patch analysis detects object removal forgery, which is always created using temporally adjacent frames, better than sequential analysis.

We compare forgery localization between sequential and patch analyses by visualizing a movement of removed objects. Fig.13 shows three examples for visualization results of a removed object movement using univariate, multivariate, and patch analyses. We observe that our patch analysis localizes the removed object movement more accurately compared to univariate and multivariate analyses, which is expected because detection results using patch analysis at the pixel level are improved as discussed in Sec.4.4.5.2. However, the false positive rate of patch analysis is higher than the false positive rates of univariate and multivariate analyses as shown in Fig.7c, which is expected because the largest Precision value in Table IVa is less than the largest Precision value in Table IIIa.

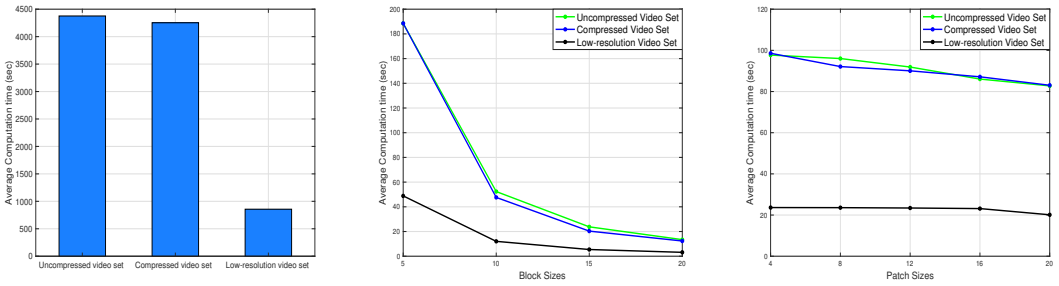
4.4.6 Computational Complexity

We will use the following notation throughout this section. N is the number of frames in a video, M is the number of pixels in each frame, B is the number of pixels in each block, and P is the number of blocks in each patch.

Univariate analysis detects additive changes that are associated with a pixel sequence in a video using the binary segmentation algorithm. We know that the computational complexity of the binary segmentation is $O(N\log(N))$ [111]. Therefore, the computational complexity of univariate analysis is $O(MN\log(N))$ because univariate analysis detects changes in each pixel. We also show the average computation time per video in seconds for the three video sets using univariate analysis in Fig.14a. We observe that the average computation time for low-resolution video set is much less than the average computation time for uncompressed and compressed video sets, which is expected because the resolution of this video set is reduced by half compared to the other video sets.

Multivariate analysis detects additive changes associated with feature vectors that are extracted from dividing video frames into non overlapping blocks. Each block requires $O(NB)$ computations to extract feature vectors throughout video frames and $O(N\log(N))$ computations to detect changes using the binary segmentation algorithm. As a result, the computational complexity of multivariate analysis is $O(M/B(NB+N\log(N)))$. We also show the average computation time per video in seconds for the three video sets using multivariate analysis in Fig.14b. We observe that the average computation time is exponentially reduced for the three video sets. The reason is that the multivariate analysis is performed for each block instead of each pixel, hence the average computation time is dramatically decreased when the block size increases.

Patch analysis detects anomalous patches, which are temporally adjacent blocks, throughout video frames by examining each patch in an overlapping manner with overlapping step equals one. The computational complexity of patch analysis is similar to multivariate analysis. The only difference is that calculating the log-likelihood of overlapping patches throughout video frames requires $O((N-P)N)$



(a) (b) (c)

Figure 14: Average computation time per video in seconds for the three video sets using (a) univariate analysis, (b) multivariate analysis with different block sizes, and (c) patch analysis with different patch sizes.

computations. As a result, the computational complexity of patch analysis is $O(M/B(NB+(N-P)N))$. We also show the average computation time per video in seconds for the three video sets using patch analysis in Fig.14c. We observe that the average computation time does not significantly change when the patch size increases because our patch analysis approach is performed based on overlapping patches instead of non overlapping patches.

We conclude that by using either the multivariate or patch analysis not only improves the detection performance compared to univariate analysis as discussed in Sec.4.4.5, but also results in much less computational time, which is another major contribution of this work.

4.4.7 Comparison Results with other Approaches

We compare our approach with five recent approaches [2, 69–72]. We refer to [2] as UniSeq, [69] as StatFeat, [70] as StegFeat, [71] as CompSen, and [72] as STCA throughout the comparison results.

Detection results at video level for our patch analysis approach and the other approaches using different video sets are shown in Table V. We set the patch size to 12 across the three video sets to have a

TABLE V: COMPARISON RESULTS OF OBJECT REMOVAL FORGERY DETECTION AT VIDEO LEVEL FOR OUR PATCH ANALYSIS APPROACH AND OTHER APPROACHES USING DIFFERENT VIDEO SETS.

Approach	Precision (%)			Recall (%)			F1 (%)			IoU(%)		
	Uncompressed	Compressed	Low-resolution	Uncompressed	Compressed	Low-resolution	Uncompressed	Compressed	Low-resolution	Uncompressed	Compressed	Low-resolution
StatFeat [69]	82.61	72.34	78.49	84.44	75.56	81.11	83.52	73.91	79.78	71.70	58.62	66.36
STCA [72]	89.66	77.78	80.00	86.67	75.27	79.12	88.14	76.50	79.56	78.79	61.95	66.06
CompSen [71]	90.80	79.51	81.91	87.78	73.34	77.78	89.21	76.30	79.79	80.52	61.68	66.37
StegFeat [70]	**97.80**	95.45	**96.59**	**98.89**	93.34	94.45	**98.34**	94.38	**95.50**	**96.73**	89.36	**91.39**
UniSeq [2]	93.33	92.22	92.86	93.33	92.22	86.67	93.33	92.22	89.66	87.49	85.56	81.26
Our approach	95.65	**95.51**	93.48	97.78	**94.44**	**95.56**	96.70	**94.97**	94.51	93.62	**90.43**	89.58

fair comparison with the other approaches. We observe that detection results of our approach are consistent across the three video sets. We also observe that StegFeat achieves slightly better performance compared to our approach throughout uncompressed and low-resolution video sets. However, our approach outperforms all five approaches in compressed video set. This result indicates that our approach is more practical because most of the online videos are compressed. Moreover, our approach not only detects forgery but also localizes forged regions, unlike other approaches [2, 69–71]

Detection results at pixel level for our patch analysis approach and the STCA approach using different video sets are shown in Tables VIa to VIc. We compare with the STCA approach only because the other approaches are not able to detect pixel level forgery. We observe that our approach outperforms the STCA approach throughout the three video sets. We also observe that detection results of our approach are consistent across the three video sets.

4.4.8 Generalization

To evaluate the generalization of our approach using different data sets and different inpainting algorithms, we use the data set that is introduced by Lin et al. [72]. This data set consists of 26 object removal

TABLE VI: COMPARISON RESULTS OF OBJECT REMOVAL FORGERY DETECTION AT PIXEL LEVEL FOR OUR PATCH ANALYSIS APPROACH AND THE STCA APPROACH USING DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF UNCOMPRESSED VIDEO SET.

Approach	Precision (%)	Recall (%)	F1 (%)	IoU (%)
STCA [72]	**78.68**	75.35	76.98	62.57
Our approach	74.49	**88.90**	**81.06**	**68.15**

(b) DETECTION RESULTS OF COMPRESSED VIDEO SET.

Approach	Precision (%)	Recall (%)	F1 (%)	IoU (%)
STCA [72]	72.70	68.26	70.41	54.33
Our approach	**75.17**	**87.08**	**80.69**	**67.63**

(c) DETECTION RESULTS OF LOW-RESOLUTION VIDEO SET.

Approach	Precision (%)	Recall (%)	F1 (%)	IoU (%)
STCA [72]	73.18	71.40	72.28	56.59
Our approach	**78.58**	**82.38**	**80.44**	**67.27**

(d) DETECTION RESULTS OF LIN'S VIDEO SET.

Approach	Precision (%)	Recall (%)	F1 (%)	IoU (%)
STCA [72]	**78.40**	72.10	75.12	60.15
Our approach	77.28	**91.87**	**83.95**	**72.33**

forged videos that are generated using two inpainting algorithms: temporal copy-and-paste [119] and exemplar-based texture synthesis [120]. This data set contains forged videos with multiple removed objects as shown in Fig.15. All videos in this dataset are compressed using MPEG-4 with 3Mbps and a resolution of 320×240. We refer to this dataset as Lin's video set throughout the comparison results.

Detection results at pixel level for our patch analysis approach and the STCA approach using Lin's video set are shown in Table VId. We observe that STCA achieves a slightly better Precision score

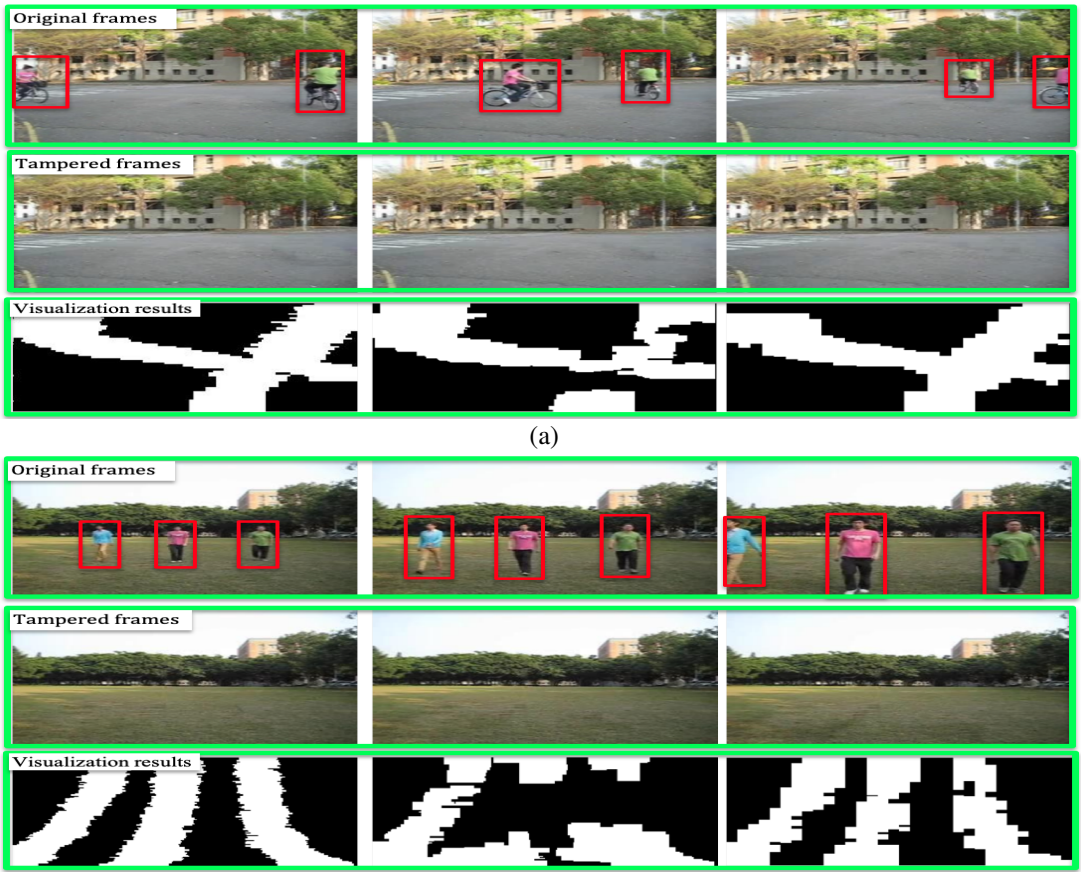


Figure 15: Two examples of visualization results of removed objects' movement using our patch analysis approach and STCA approach. In each example, images on the top row indicate frames from the original video; images on the middle row indicate the corresponding frames from the tampered video where the objects in the red box have been removed from the scene; images on the bottom row (from left to right) indicate the ground truth of the removed objects' movement, the movement visualization using the STCA approach, and our patch analysis approach, respectively.

compared to our approach. However, our approach outperforms STCA in terms of Recall, F1, and IoU scores. This result confirms that our approach can detect and localize object removal forgery in forged videos with multiple removed objects for different data sets and inpainting algorithms.

Two examples of localization results for our patch analysis approach and the STCA approach are shown in Fig.15. We observe that our patch analysis localizes the removed objects' movement more accurately compared to STCA. In fact, our patch analysis correctly localizes three removed objects in Fig.15b. However, the STCA localizes only one removed object in Fig.15b. We believe that our patch analysis can detect and localize multiple removed objects because patch analysis detects all anomalous patches of a particular block by investigating all overlapping patches of this block as shown in Fig.11. For example, if there are two removed objects that pass through a block in video frames, then patch analysis would detect two anomalous segments for this block.

CHAPTER 5

FACEMD: CONVOLUTIONAL NEURAL NETWORK-BASED SPATIOTEMPORAL FUSION FACIAL MANIPULATION DETECTION

In this chapter, we study the facial manipulation problem, particularly facial identity and facial expression manipulations, as shown in Fig.16, and propose a novel approach, dubbed FaceMD, based on fusing three streams of convolutional neural networks to detect facial manipulation. The proposed FaceMD incorporates spatiotemporal information by fusing video frames, motion residuals, and 3D gradients. We select motion residuals and 3D gradients in addition to video frames because motion residuals and 3D gradients expose spatiotemporal artifacts that are created during video manipulation. The exposure of these spatiotemporal artifacts improves facial manipulation detection accuracy. We combine the three streams using different fusion methods and places to best use the spatiotemporal information, hence increasing detection accuracy.

5.1 Contributions

The contributions of this chapter can be summarized as follows:

1. For the first time, we apply spatiotemporal analysis to detect facial manipulation videos.
2. We analyze motion residuals and 3D gradients of video frames, and experimentally show that they improve facial manipulation detection accuracy.
3. We adapt the Xception model with multiple inputs for facial manipulation problems to enhance detection accuracy.

(a)

(b)

Figure 16: Two types of facial manipulation. (a) Facial identity manipulation using FaceSwap and DeepFakes approaches. (b) Facial expression manipulation using Face2Face and NeuralTextures approaches.

4. Our proposed FaceMD achieves a new state-of-the-art performance on FaceForensics++ [76] and Google-JigSaw [121] data sets.

The rest of this chapter is organized as follows: Our proposed FaceMD is described in Sec. 5.2. The motion residual extraction is presented in Sec. 5.2.1. The 3D gradient extraction is described in Sec. 5.2.2. The network architecture is provided in Sec. 5.2.3. The fusion methods are explained in Sec. 5.2.4. The experimental results are discussed in Sec. 5.3.

5.2 Spatiotemporal Fusion Facial Manipulation Detection

We briefly describe our proposed FaceMD in the following steps. First, We manually extract a face region from each frame to eliminate miss detection due to face-tracking failure. Then, we extract motion residual and 3D gradient frames (streams) from the input video to expose spatiotemporal variations between adjacent frames. We use video frames, motion residuals, and 3D gradients as inputs of a convolutional neural network to guide the network to find discriminative features from the spatiotemporal information. We combine these three streams using different fusion methods and places to best use the spatiotemporal information, hence increasing detection accuracy. We detect facial manipulation based on the output prediction of the network.

5.2.1 Motion Residual Extraction

In general, video frames are divided into local temporal windows. These local temporal windows contain strongly correlated frames. Each frame in these local temporal windows consists of a static part and a motion part. The static part is identical to the first frame in a local temporal window, whereas the motion part represents a motion residual relative to the first frame in this local temporal window. The motion residual contains temporal information because it shows temporal variations between adjacent frames. As a result, the motion residual becomes one of our main analyses to expose temporal inconsistency that is created during video manipulation. In practice, a local temporal window is presented by a Group Of Pictures (GOP). This GOP consists of three types of frames: I-frames (Intra coded frames), P-frames (Predictive coded frames), and B-frames (Bipredictive coded frames). The I-frame represents the first frame in each local temporal window, while P-frame and B-frame are motion residuals relative to the I-frames. However, the GOPs have a flexible structure in advanced video coding, hence it is not

feasible to use P-frame and B-frame as motion residuals. Therefore, we use collision operators to extract motion residuals in our approach [70, 122, 123]. Let us assume that a video clip consists of N frames as follows:

$$V = \{F^{(1)}, F^{(2)}, \ldots, F^{(N)}\}, F_{i,j}^{(k)} \in [0, 255]^3 \tag{5.1}$$

where $F^{(k)}$ represents the kth video frame that is an 8-bit RGB image with size $m \times n \times 3$. Then, a mean collision is defined as follows:

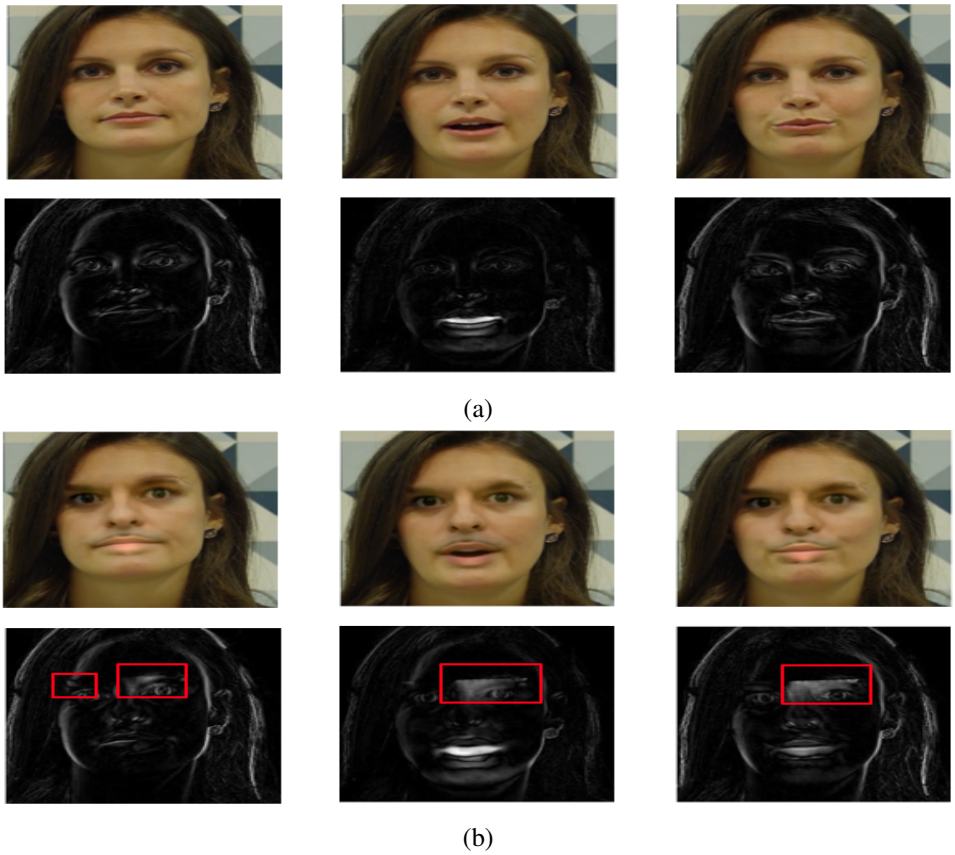
$$C^{(k)} = \sum_{n=k-l}^{k+l} \frac{F^{(n)}}{2l-1} \tag{5.2}$$

where $C^{(k)}$ is the colluded result of a local temporal window that is centered at frame $F^{(k)}$, and l is the number of left (right) neighbors of $F^{(k)}$. In our experiment, we set this number to 9. Then, the motion residual of $F^{(k)}$ becomes

$$R^{(k)} = |F^{(k)} - C^{(k)}| \tag{5.3}$$

where $|\cdot|$ is the absolute value. Note that $F_{i,j}^{(k)} - C_{i,j}^{(k)} \in [-255, 255]^3$, hence the absolute value is essential to keep the motion residual values between 0 and 255. We extract motion residual from red, green, and blue channels of video frames, hence the size of the motion residual frame is $m \times n \times 3$.

Video manipulation creates independent facial manipulation frames, as discussed in Sec.2.2.2. This manipulation would cause temporal inconsistency between adjacent frames. This temporal inconsis-



(a)

(b)

Figure 17: Sample frames from original and manipulated videos with the corresponding frames from motion residuals. (a) Three arbitrary frames from an original video and the corresponding motion residual frames. (b) Three arbitrary frames from a facial identity manipulation video and the corresponding motion residual frames where the red boxes show temporal artifacts caused by facial manipulation.

tency can be exposed by extracting motion residual frames. In particular, we experimentally observe that motion residual shows abnormal motion when a video frame is manipulated, as shown in Fig.17. This figure (Fig.17a) shows three arbitrary frames from an original video and the corresponding motion residual frames. These motion residual frames show normal motion that appears around eyes, nose, and mouth regions. The other figure (Fig.17b) shows three arbitrary frames from a facial identity manip-

ulation video and the corresponding motion residual frames. The red boxes in motion residual frames show abnormal motion caused by video manipulation. These abnormal motion appear above the eyes and nose regions. This result indicates that the motion residual can expose facial manipulation artifacts. Therefore, the motion residual becomes one of our main analyses to detect facial manipulation.

5.2.2 3D Gradient Extraction

A 3D gradient is a directional change in the intensity of video frames. The gradient in x- and y-directions measures horizontal and vertical changes in the intensity, respectively, while the gradient in z-direction measures temporal changes in the intensity. These changes represent edges in the x-, y-, and z-directions, hence the 3D gradient detects spatial and temporal edges (changes). As a result, the 3D gradient becomes one of our main analyses to expose spatiotemporal changes (artifacts) that are created during video manipulation. We use a simple but effective Sobel gradient operator [124]. This operator convolves a 3-by-3-by-3 kernel with video frames to approximate the 3D gradient. The 3D gradient of a video frame is as follows:

$$\nabla F(x,y,z) = \begin{bmatrix} \frac{\partial F}{\partial x} \\ \frac{\partial F}{\partial y} \\ \frac{\partial F}{\partial z} \end{bmatrix} = \begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix} \tag{5.4}$$

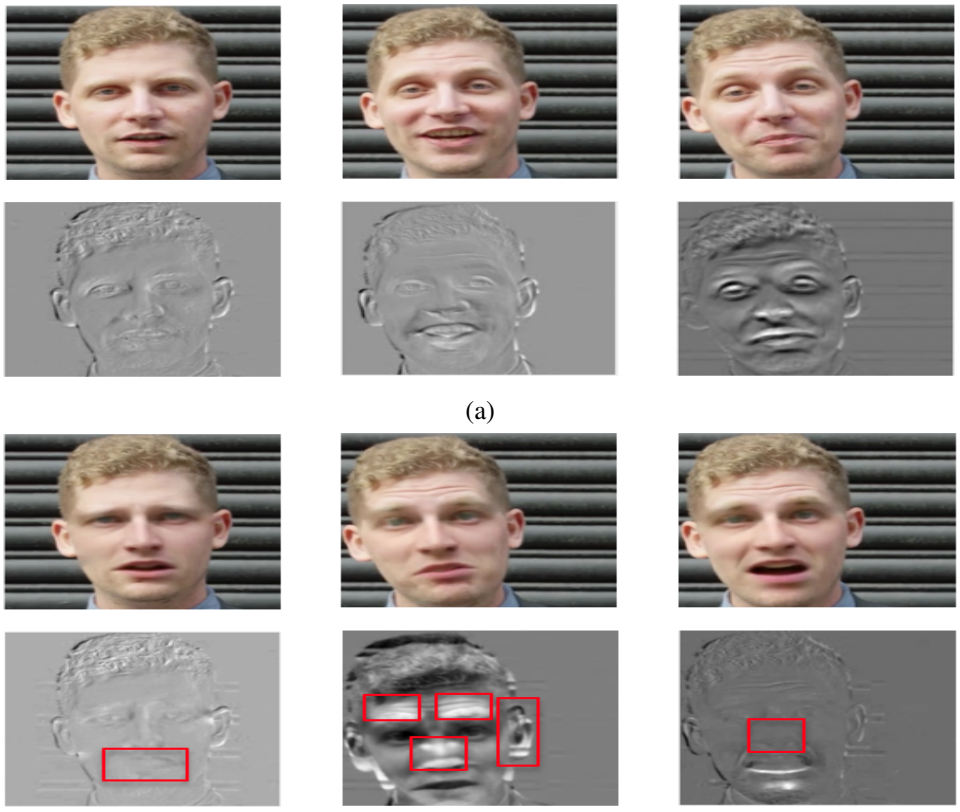
where G_x, G_y, and G_z are given by

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * F(x,y,z-1) + \begin{bmatrix} 2 & 0 & -2 \\ 4 & 0 & -4 \\ 2 & 0 & -2 \end{bmatrix} * F(x,y,z) + \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * F(x,y,z+1) \quad (5.5)$$

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * F(x,y,z-1) + \begin{bmatrix} 2 & 4 & 2 \\ 0 & 0 & 0 \\ -2 & -4 & -2 \end{bmatrix} * F(x,y,z) + \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * F(x,y,z+1) \quad (5.6)$$

$$G_z = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} * F(x,y,z-1) + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} * F(x,y,z) + \begin{bmatrix} -1 & -2 & -1 \\ -2 & -4 & -2 \\ -1 & -2 & -1 \end{bmatrix} * F(x,y,z+1) \quad (5.7)$$

where $F(x,y,z)$ represents the zth video frame that is an 8-bit gray-scale image with size $m \times n$; G_x, G_y, and G_z represent the gradient in x-, y- and z-directions, receptively; $*$ represents the convolution operation. We stack G_x, G_y, and G_z in the channel dimension, hence the size of the 3D gradient frame is $m \times n \times 3$.



(a)

(b)

Figure 18: Sample frames from original and manipulated videos with the corresponding frames from 3D gradient in the z-direction. (a) Three arbitrary frames from an original video and the corresponding 3D gradient frames in the z-direction. (b) Three arbitrary frames from a facial expression manipulation video and the corresponding 3D gradient frames in the z-direction where the red boxes show spatiotemporal artifacts caused by facial manipulation..

Facial manipulation causes spatiotemporal artifacts, as discussed in Sec.2.2.2. These artifacts can be exposed by extracting 3D gradient frames. In particular, we experimentally observe that when a video frame is manipulated, the 3D gradient either doesn't detect edges at expected edge regions or detects edges at unexpected edge regions, as shown in Fig.18. This figure (Fig.18a) shows three arbitrary frames

from an original video and the corresponding 3D gradient frames in the z-direction. These 3D gradient frames detect all edges at expected edge regions. The other figure (Fig.18b) shows three arbitrary frames from a facial expression manipulation video and the corresponding 3D gradient frames in the z-direction. We observe that edges at mouth and nose regions are not detected, as shown in the first and third gradient frames due to blurring artifacts. We also observe large gradient values above the eyes, ears, and nose regions in the second gradient frame, which is not expected. This result indicates that the 3D gradient can expose facial manipulation artifacts. Therefore, the 3D gradient becomes one of our main analyses to detect facial manipulation.

5.2.3 Network Architecture

We could use 3D Convolutional Neural Networks (3D ConvNets) for video manipulation detection because these networks seem to be a natural approach for this manipulation. However, these 3D ConvNets have the following limitations. First, the 3D ConvNets require large memory and more GPUs because these networks take as an input a large batch of frames; hence these networks need more computational resources that are not always available [125]. Second, the 3D ConvNets have significantly more parameters than 2D ConvNets; therefore training the 3D ConvNets requires a large data set. Furthermore, the 3D ConvNets have shown promising results on video classification problems, but these results are not competitive with state-of-the-art results [126]. As a result, We adapt the Xception model, which is a 2D ConvNets, with three inputs for facial manipulation detection [127]. The three inputs are video frames that represent spatial information, motion residual frames that represent temporal information, and 3D gradient frames that represent spatiotemporal information.

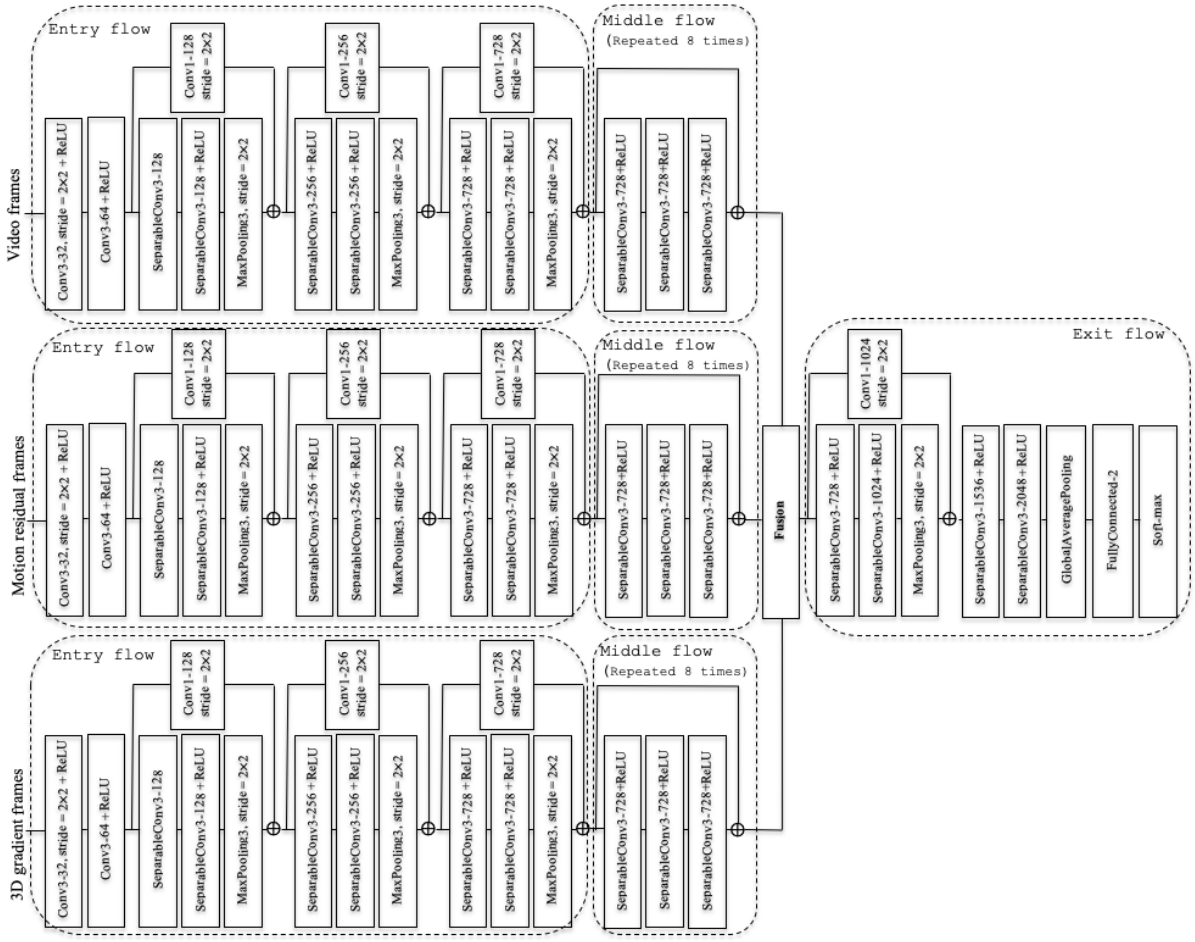


Figure 19: The proposed architecture to facial manipulation detection.

The adapted Xception architecture is shown in Fig.19. We notate layers in this figure as "{layer's type}{kernel size}-{number of channels}". For example, "conv3-32" represents a convolutional layer with kernel size equals 3x3, and the number of channels is 32. As can be seen from Fig.19, each input goes through the entry and middle flows that consist of 31 convolutional layers to extract features from each of the three inputs. Then, these three inputs are combined using different fusion methods that

will be introduced in Sec. 5.2.4. The resultant steam goes through the exit flow that consists of five convolutional layers, followed by a fully connected layer whose output is a two-dimensional vector. Note that all convolutional layers are followed by batch normalization.

5.2.4 Fusion Methods

We use different fusion functions to fuse three feature maps coming from three streams that are video frames, motion residuals, and 3D gradients [128]. We fuse these three streams at a particular layer by combining feature maps at the same pixel location in the three streams. We perform the fusion to guide the ConvNet to find discriminative features from spatial (video frames), temporal (motion residuals), and spatiotemporal (3D gradients) information. In general, a fusion function fuses N feature maps as follows:

$$f : x^{(1)}, x^{(2)}, \ldots, x^{(N)} \rightarrow y \tag{5.8}$$

where $x^{(n)} \in R^{H_n \times W_n \times D_n}$ and $y \in R^{H \times W \times D}$; H, W, and D are the height, width, and the number of channels of a particular feature map, respectively. For simplicity we assume that $H_n = H$, $W_n = W$, $D_n = D$. Then, we define the sum, average, max, and concatenate fusions as follows:

5.2.4.1 Sum Fusion

This function calculates the sum of N feature maps at the same spatial and channel locations as follows:

$$y_{i,j,k} = \sum_{n=1}^{N} x_{i,j,k}^{(n)} \tag{5.9}$$

where $i \leq 1 \leq H$, $j \leq 1 \leq W$, $k \leq 1 \leq D$ and $x^{(n)}, y \in R^{H \times W \times D}$.

5.2.4.2 Average Fusion

This function computes the average of N feature maps at the same spatial and channel locations as follows:

$$y_{i,j,k} = \frac{1}{N} \sum_{n=1}^{N} x_{i,j,k}^{(n)} \tag{5.10}$$

where all variables are defined as in (5.9).

5.2.4.3 Max Fusion

This function returns the largest value in N feature maps at the same spatial and channel locations as follows:

$$y_{i,j,k} = \max_{1 \leq n \leq N} \{x_{i,j,k}^{(n)}\} \tag{5.11}$$

where all variables are defined as in (5.9).

5.2.4.4 Concatenate Fusion

This function stacks N feature maps at the same spatial locations across the channels as follows:

$$y_{i,j,N \times k} = [x_{i,j,k}^{(1)}; x_{i,j,k}^{(2)}; \ldots; x_{i,j,k}^{(N)}] \tag{5.12}$$

where $y \in R^{H \times W \times ND}$ and the other variables are defined as in (5.9).

In the next section, we will evaluate the performance of these four fusions. We will also evaluate our model at different fusion layers representing early-fusion, mid-fusion, and late-fusion. The early-fusion represents fusing the three streams after the entry flow. The mid-fusion represents fusing the three streams after the middle flow. The late-fusion represents fusing the three streams after the exit flow (the last convolution layer).

5.3 Experimental analysis

In this section, we describe facial manipulation data sets and evaluation measurements. We also present implementation settings and analyze the detection results of the proposed FaceMD. Then, we compare FaceMD with state-of-the-art approaches.

5.3.1 Facial Manipulation Data Sets

We conduct our experiments on FaceForensics++ [76] and Google-JigSaw [121] data sets. The FaceForensics++ consists of 1000 original videos that have been manipulated with four facial identity and facial expression manipulations. The facial identity manipulation is created using FaceSwap [8] and DeepFakes [9]. The facial identity expression is created using Face2Face [10] and NeuralTextures [11]. All manipulation videos are categorized into four groups based on manipulation types. Therefore, this data set evaluates our approach to detect specific types of facial manipulation. The Google-JigSaw contains 3000 manipulated videos that are recorded from 28 actors in different scenes. These manipulated videos are created using different facial identity and facial expression manipulations without categorizing these videos based on manipulation types. Therefore, this data set evaluates our approach to detect general types of facial manipulation.

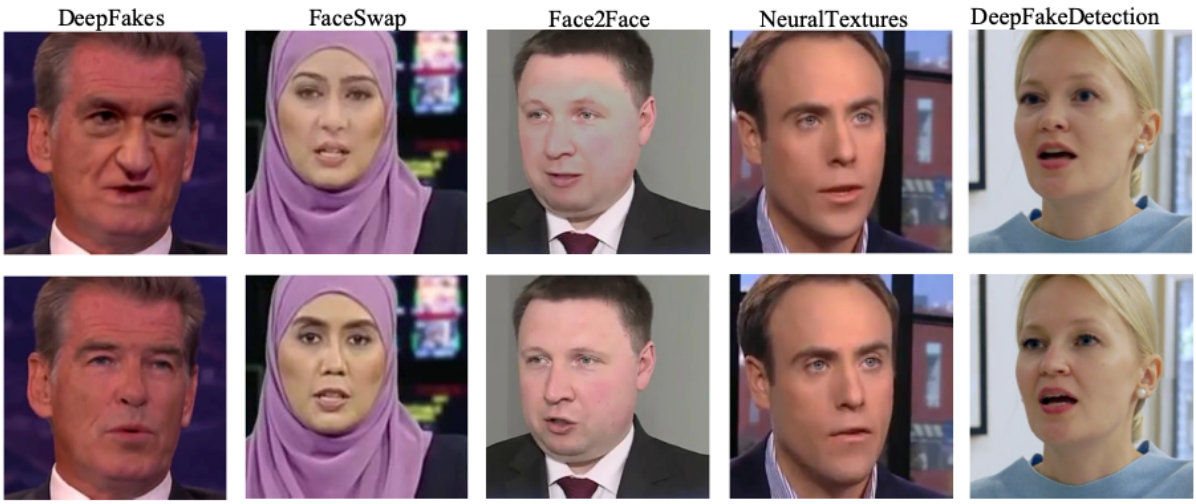


Figure 20: A sample frame from each category of facial manipulation in the two data sets: Images on the top row indicate sample frames from each type of facial manipulation; Images on the bottom row indicate corresponding frames from the original video.

The FaceForensics++ and Google-JigSaw data sets consist of original and manipulated videos with different qualities that are raw, high-quality, and low-quality to represent real-world videos. The raw videos represent videos without compression. The high-quality videos represent videos that have been compressed using the H.264 codec with a quantization equals 23. The low-quality videos represent videos that have been compressed using the H.264 codec with a quantization equals 40.

We randomly select 100 videos that contain 100 frames from each category in the FaceForensics++ and Google-JigSaw data sets because we have limited computational resources. We also select these 100 videos to have the same number of videos in each category since each category has a different number of videos. The FaceForensics++ has five categories that are original, DeepFakes, FaceSwap, Face2Face, and NeuralTextures, hence we have 500 videos that contain 50,000 frames for each video

quality. The Google-JigSaw has two categories that are original and DeepFakeDetection; thus we have 200 videos that contain 20,000 frames for each video quality. As a result, we have 210,000 frames to evaluate our approach using the FaceForensics++ and Google-JigSaw data sets. Fig.20 shows a sample frame from each category of facial manipulation in these two data sets. We split these data sets into 60% for training, 10% for validation, and 30% for testing.

5.3.2 Evaluation Metric

We evaluate facial manipulation detection based on frame classification. We measure the performance by defining T_P as the correctly detected forged frames, T_N as the correctly identified original frames, F_P as original frames that have been incorrectly detected as forged, and F_N as falsely missed forged frames. Then, *Accuracy* is as follows:

$$Accuracy = \frac{T_P + T_N}{T_P + T_N + F_P + F_N} \tag{5.13}$$

The *Accuracy* measures how many frames, both original and fake, are correctly classified. We also evaluate the performance of facial manipulation detection using Area Under the Curve (*AUC*). The *AUC* measures the trade-off between true positive and false positive rates.

5.3.3 Implementation Settings

We implement our model using the TensorFlow framework [129] and train our model on a NVIDIA P100 GPU. We train our model using a batch size equals 10 with a frame size of $299 \times 299 \times 3$. We use the Stochastic Gradient Descent (SGD) optimizer with a constant learning rate of 0.001 and a momen-

TABLE VII: DETECTION RESULTS OF THE LOW-QUALITY DEEPFAKES USING DIFFERENT FUSION METHODS.

Fusion method	Accuracy (%)	AUC (%)	Parameter count
Sum	**85.35**	**94.69**	48,848,186
Average	83.50	91.26	48,848,186
Max	77.65	86.47	48,848,186
Concatenate	80.75	88.52	**51,412,202**

tum of 0.9. We train our model for 40 epochs and choose the best model based on the validation set. The detection results are reported using the test set.

5.3.4 Detection Analysis

We select the low-quality DeepFakes to analyze the detection results of our model using different fusion methods and places. We select this low-quality category to evaluate our model on the worst quality video set.

5.3.4.1 Detection Results of Fusion Methods

Detection results of the low-quality DeepFakes using different fusion methods are shown in Table VII. We observe that Sum and Average fusions perform better than Max and Concatenate fusions. Sum fusion achieves the highest detection accuracy and AUC with less number of parameters compared to Concatenate fusion. Sum, Average, and Max fusions have the same number of parameters because these fusions do not increase the number of output channels, whereas Concatenate fusion increases the number of output channels.

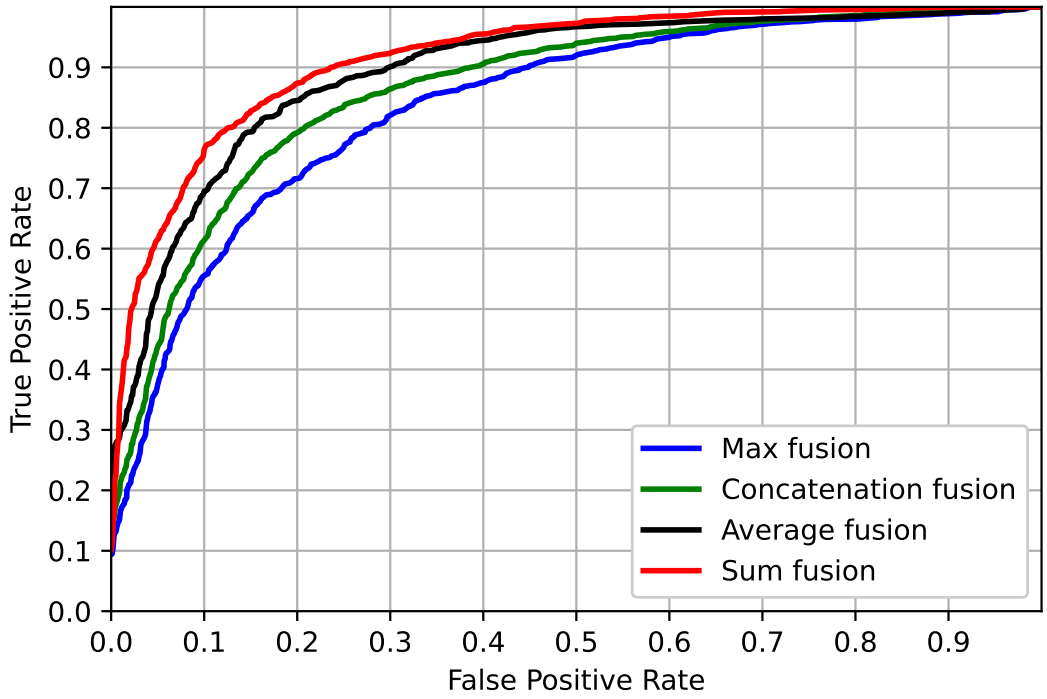


Figure 21: ROC curves: True positive vs. false positive rates for different fusion methods using the low-quality DeepFakes video set.

We also compare facial manipulation detection of different fusion methods by plotting receiver operating characteristic (ROC) curves, as shown in Fig.21. We observe that Max fusion achieves the lowest detection performance. We think the reason is that max fusion returns the largest value of the three streams, hence this fusion considers only one (the largest) stream in the following layers. Sum fusion achieves the highest detection performance that is slightly better than the detection performance of Average fusion. Therefore, we use Sum fusion in our model to combine the three streams.

TABLE VIII: DETECTION RESULTS OF THE LOW-QUALITY DEEPFAKES USING DIFFERENT FUSION PLACES.

Fusion place	Accuracy (%)	AUC (%)	Parameter count
Early-fusion	82.30	90.39	23,024,570
Mid-fusion	**85.35**	**94.69**	48,848,186
Late-fusion	82.74	90.84	**62,424,954**

5.3.4.2 Detection Results of Fusion Places

We evaluate our model at different fusion places representing early-fusion, mid-fusion, and late-fusion. The early-fusion represents fusing the three streams after the entry flow. The mid-fusion represents fusing the three streams after the middle flow. The late-fusion represents fusing the three streams after the exit flow (the last convolution layer). Detection results of the low-quality DeepFakes using different fusion places are shown in Table VIII. We observe that mid-fusion achieves the highest detection accuracy and AUC, which suggests that using a deeper network before the fusion enhances detection performance. However, late-fusion achieves relatively low detection performance. We think the reason is that the convolution layers after the fusion are essential to extract features from the resultant steam. We also observe that late-fusion has the highest number of parameters, which is expected because its network before the fusion is deeper than the networks before the fusion of early- and mid-fusions.

We also compare facial manipulation detection of different fusion places by plotting receiver operating characteristic (ROC) curves, as shown in Fig.22. We observe that early-fusion achieves detection performance similar to the detection performance of late-fusion. Mid-fusion achieves the highest detec-

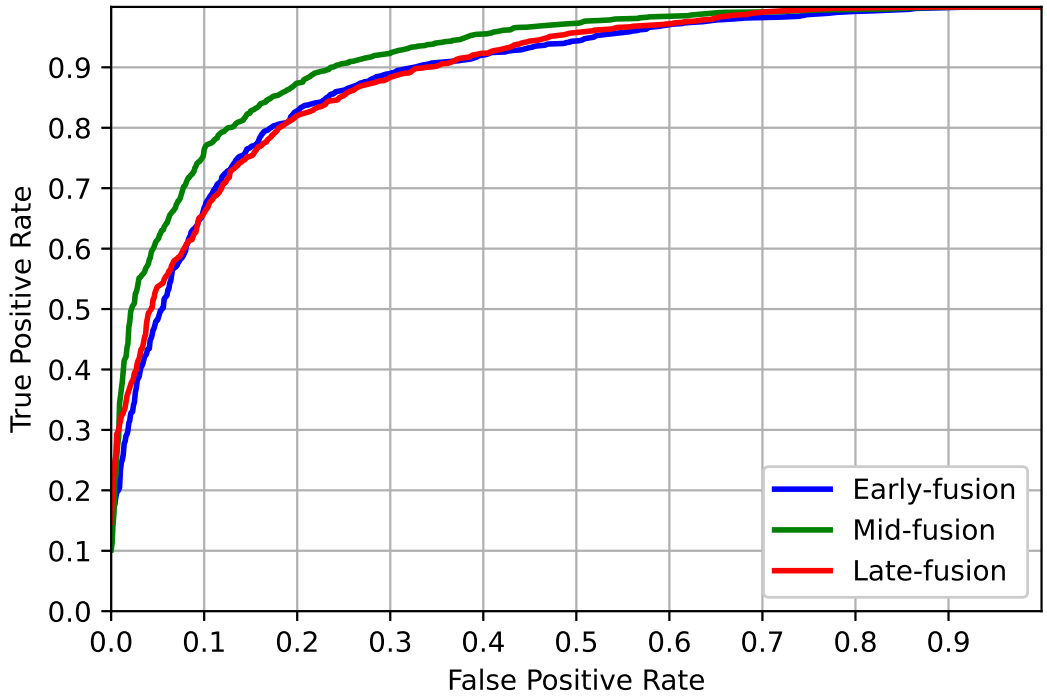


Figure 22: ROC curves: True positive vs. false positive rates for different fusion places using the low-quality DeepFakes video set.

tion performance because it consists of a deep network before the fusion to extract features from each stream, followed by five convolution layers to extract features from the resultant steam. As a result, we combine the three streams after the middle flow that is mid-fusion, as shown in Fig.19.

5.3.5 Comparison Results with other Approaches

We compare our proposed FaceMD with three recent approaches: Xception [127], MesoNet [89], and Two-stream [91]. We compare with Xception and MesoNet approaches because these approaches

achieve the highest detection performance using FaceForensics++ data set [76]. We compare with the Two-stream approach because this approach uses two streams of neural networks, hence it is similar to our approach. However, these streams are based on spatial information. We evaluate the proposed FaceMD and the other approaches using the sub data sets that are discussed in Sec. 5.3.1. We also evaluate these approaches to detect specific and general types of facial manipulation.

5.3.5.1 Specific Manipulation Detection

We evaluate our proposed FaceMD and the other approaches to detect each type of facial identity manipulation and facial expression manipulation. We use FaceForensics++ data set to evaluate specific manipulation detection because this data set contains manipulated videos that are categorized into four groups based on manipulation types.

5.3.5.1.1 Detection of Facial Identity Manipulation

We evaluate our proposed FaceMD and the other approaches to detect DeepFakes and FaceSwap manipulations.

Detection results of DeepFakes manipulation for our proposed FaceMD and the other approaches using different video qualities are shown in Table IX. We observe that MesoNet and Two-stream have similar detection performance that is less than the detection performance of Xception. The FaceMD achieves the highest detection accuracy and AUC throughout the three video qualities. We also observe that the FaceMD achieves the highest detection performance using the raw video set, and then it decreases throughout the high-quality and low-quality video sets. This result indicates that motion

TABLE IX: COMPARISON RESULTS OF DEEPFAKES DETECTION FOR OUR PROPOSED FACEMD AND OTHER APPROACHES USING DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF RAW VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	89.60	96.94
Two-stream [91]	87.26	95.68
MesoNet [89]	86.94	95.23
Proposed FaceMD	**93.80**	**98.31**

(b) DETECTION RESULTS OF HIGH-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	84.90	93.75
Two-stream [91]	80.63	88.33
MesoNet [89]	81.19	90.40
Proposed FaceMD	**88.48**	**96.32**

(c) DETECTION RESULTS OF LOW-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	80.70	90.29
Two-stream [91]	75.46	85.12
MesoNet [89]	77.51	86.83
Proposed FaceMD	**85.35**	**94.69**

residual and 3D gradient streams improve detection results of DeepFakes videos with different qualities.

Detection results of FaceSwap manipulation for our proposed FaceMD and the other approaches using different video qualities are shown in Table X. We observe that the FaceMD achieves less detection performance compared to the detection performance of the FaceMD using DeepFakes. We also observe that detection accuracy and AUC of the FaceMD decrease when video quality decreases. However, the FaceMD outperforms the other approaches throughout the three video qualities of FaceSwap manipulation. This result confirms that the extracted features from the spatiotemporal streams enhance detection accuracy and AUC of FaceSwap videos with different qualities.

In summary, our proposed FaceMD outperforms the other approaches using the two types of facial identity manipulation with significantly high detection accuracy and AUC. We observe that the AUC of FaceMD is decreased by only 4% when using the low-quality video set of the two types of facial identity manipulation. Hence, the FaceMD is robust against different video qualities of facial identity manipulation. We believe that the FaceMD is robust for two reasons. First, the adapted model is able to extract discriminative features even though videos have low-quality. Second, reducing video quality might conceal spatial artifacts, but it will not conceal temporal artifacts that are exposed by extracting motion residual frames. We also observe that combining video frames, motion residuals, and 3D gradients improves detection performance compared to Xception that is the backbone of our model.

5.3.5.1.2 Detection of Facial Expression Manipulation

We evaluate our proposed FaceMD and the other approaches to detect Face2Face and NeuralTextures manipulations.

TABLE X: COMPARISON RESULTS OF FACESWAP DETECTION FOR OUR PROPOSED FACEMD AND OTHER APPROACHES USING DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF RAW VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	88.32	95.22
Two-stream [91]	85.76	93.41
MesoNet [89]	83.91	92.73
Proposed FaceMD	**91.62**	**96.92**

(b) DETECTION RESULTS OF HIGH-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	83.24	92.36
Two-stream [91]	80.13	89.50
MesoNet [89]	79.43	87.84
Proposed FaceMD	**87.56**	**95.78**

(c) DETECTION RESULTS OF LOW-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	79.68	88.15
Two-stream [91]	76.33	85.47
MesoNet [89]	74.92	82.66
Proposed FaceMD	**82.84**	**91.20**

Detection results of Face2Face manipulation for our proposed FaceMD and the other approaches using different video qualities are shown in Table XI. We observe that the Two-stream approach achieves

TABLE XI: COMPARISON RESULTS OF FACE2FACE DETECTION FOR OUR PROPOSED FACEMD AND OTHER APPROACHES USING DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF RAW VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	85.12	92.88
Two-stream [91]	83.76	91.57
MesoNet [89]	83.07	91.18
Proposed FaceMD	**86.98**	**94.92**

(b) DETECTION RESULTS OF HIGH-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	82.54	90.73
Two-stream [91]	79.36	87.66
MesoNet [89]	78.83	86.97
Proposed FaceMD	**84.42**	**92.39**

(c) DETECTION RESULTS OF LOW-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	79.62	88.50
Two-stream [91]	75.95	85.34
MesoNet [89]	75.27	84.71
Proposed FaceMD	**80.89**	**89.08**

detection performance slightly better than the detection performance of MesoNet. The difference between the detection performance of FaceMD and detection performance of Xception becomes small because this manipulation is hard to detect compared to facial identity manipulation. Even though the difference is small, the FaceMD outperforms the other approaches, including Xception, throughout the three video qualities. This result indicates that combining spatiotemporal streams with video frames stream improves detection results of Face2Face videos with different qualities.

Detection results of NeuralTextures manipulation for our proposed FaceMD and the other approaches using different video qualities are shown in Table XII. We observe that Xception achieves slightly better detection performance compared to FaceMD using the low-quality video set. The FaceMD achieves slightly less detection accuracy and AUC because this manipulation is harder than Face2Face manipulation, and this video set has the lowest video quality. However,the FaceMD outperforms the other approaches, including Xception, throughout the raw and high-quality video sets. This result indicates that motion residual and 3D gradient streams improve detection results of NeuralTextures videos that have high-quality.

In summary, our proposed FaceMD detects the two types of facial expression manipulation with high detection accuracy and AUC. We observe that the FaceMD achieves less detection performance compared to the detection performance of the FaceMD using facial identity manipulation, which is expected because facial expression manipulation only replaces the facial expression while the facial identity remains the same. As a result, it is hard to detect facial expression manipulation. Even though the detection performance of the FaceMD decreases when video quality decreases, it outperforms the other approaches throughout the raw and high-quality video sets of the two types of facial expression

TABLE XII: COMPARISON RESULTS OF NEURALTEXTURES DETECTION FOR OUR PROPOSED FACEMD AND OTHER APPROACHES USING DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF RAW VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	80.39	88.23
Two-stream [91]	76.82	85.66
MesoNet [89]	74.65	82.75
Proposed FaceMD	**81.46**	**90.58**

(b) DETECTION RESULTS OF HIGH-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	78.24	87.60
Two-stream [91]	73.51	81.45
MesoNet [89]	71.33	79.29
Proposed FaceMD	**78.86**	**88.12**

(c) DETECTION RESULTS OF LOW-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	**72.78**	**82.85**
Two-stream [91]	65.16	75.73
MesoNet [89]	64.81	75.28
Proposed FaceMD	71.63	80.47

manipulation. This result confirms that fusing spatiotemporal information that is extracted from video frames enhances the detection performance of facial expression manipulation.

5.3.5.2 General Manipulation Detection

We evaluate our proposed FaceMD and the other approaches to detect general types of facial manipulation. We know that the most important aspect in practice is to determine whether a video is manipulated or not. Therefore, we evaluate the ability of FaceMD to discriminate between fake and original videos regardless of the type of facial manipulation. We use Google-JigSaw (DeepFakeDetection) data set to evaluate general manipulation detection because this data set contains manipulated videos that are created using different facial identity and facial expression manipulations without categorizing these videos based on manipulation types.

Detection results of DeepFakeDetection manipulation for our proposed FaceMD and the other approaches using different video qualities are shown in Table XIII. We observe that the FaceMD achieves the highest detection performance using the raw video set, and then decreases throughout the high-quality and low-quality video sets. The FaceMD outperforms the other approaches throughout the three video qualities. We believe that the FaceMD achieves the highest detection performance because fusing spatiotemporal information guides our model to find discriminative features that improve detection performance. This result confirms that our proposed FaceMD is able to correctly identify manipulated frames without knowing specific types of facial manipulation.

TABLE XIII: COMPARISON RESULTS OF GENERAL MANIPULATION DETECTION FOR OUR PROPOSED FACEMD AND OTHER APPROACHES USING DIFFERENT VIDEO SETS.

(a) DETECTION RESULTS OF RAW VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	86.76	94.26
Two-stream [91]	82.48	90.85
MesoNet [89]	81.17	88.32
Proposed FaceMD	**88.83**	**95.41**

(b) DETECTION RESULTS OF HIGH-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	83.24	91.52
Two-stream [91]	80.46	88.19
MesoNet [89]	78.73	87.64
Proposed FaceMD	**85.31**	**93.80**

(c) DETECTION RESULTS OF LOW-QUALITY VIDEO SET.

Approach	Accuracy (%)	AUC (%)
Xception [127]	76.70	85.84
Two-stream [91]	75.92	83.31
MesoNet [89]	73.68	81.57
Proposed FaceMD	**80.54**	**88.26**

Figure 23: Sample frames from Peele's video and Hader's video: Images on the top row indicate a sample frame from Peele's video where he ventriloquizes Obama, followed by three sample frames from the manipulated video; Images on the bottom row indicate a sample frame from Hader's video where he talks to Letterman, followed by three sample frames from the manipulated video.

5.3.6 Real-World Cases

We evaluate our proposed FaceMD on two real-world cases. The first case was made by Jordan Peele's production company in 2018. In this video, Peele makes former US president Barack Obama voice his opinion on different topics by synthesizing Obama's mouth. The second case was uploaded to the YouTube channel Ctrl Shift Face in 2019. This video is a clip from a conversation between Bill Hader and David Letterman on his Late Show in 2008. In this video, Hader's face shifts into Seth Rogen's face when Hader is doing an impression of Rogen. Fig.23 shows sample frames from the two manipulated videos.

We observe that our proposed FaceMD detects facial manipulation of Peele's video with an accuracy of 72.18%. This result confirms that the FaceMD is able to identify facial manipulation even though this video is only manipulated by synthesizing the mouth region while the other face regions remain the

same. We also observe that our proposed FaceMD detects facial manipulation of Hader's video with an accuracy of 54.39%. The detection accuracy is low because this video is attacked by facial identity and facial expression manipulations at the same time without leaving visible traces. Furthermore, this video has a lower quality than the original video. Therefore, detecting such a manipulated video with high accuracy becomes very unlikely for any facial manipulation detection approach.

CHAPTER 6

CONCLUSION AND FUTURE WORK

We investigated digital media forgery and proposed novel approaches to detect and localize digital media forgery. Chapter 3 presented a novel approach based on the sparse representation of keypoint descriptors to detect image forgery. We proposed a new matching criterion that is performed using dictionary atoms instead of ratios between SIFT descriptors. By using this matching criterion, we eliminate efforts of adjusting a threshold. The experimental results show that our approach not only outperforms the-state-of-the-art approaches, but it is also efficient and robust against compression and rotation attacks.

Chapter 4 presented a novel approach based on sequential and patch analyses to detect video forgery and localize forged regions by visualizing a movement of removed objects. We modeled video sequences as stochastic processes, where changes in the parameters of these processes indicate a video forgery. We also modeled video sequences as a mixture model of normal and anomalous patches, with the aim to separate these patches by identifying the distribution of each patch. We evaluated detection performance at pixel and video levels, unlike most of the existing approaches that evaluated detection performance at video level only without localizing forged regions. The experimental results show that the detection performance is improved by using multivariate sequential analysis compared to univariate sequential analysis. Furthermore, our patch analysis approach not only achieves excellent detection performance with low computational complexity, but also leads to robust results against compressed and lower resolution videos.

Chapter 5 presented a novel approach (FaceMD) based on fusing three streams of convolutional neural networks to detect facial manipulation. We incorporate spatiotemporal information by fusing three streams that are video frames, motion residuals, and 3D gradients. We combine these three streams using different fusion methods and places to best use the spatiotemporal information, hence increasing detection accuracy. The experimental results show that the detection performance is significantly improved by using spatiotemporal information of video frames. Furthermore, our proposed FaceMD outperforms state-of-the-art approaches to detecting both specific and general types of facial manipulation.

In the future, we will investigate non-additive change models such as changes in covariance or correlations using the asymptotic local hypotheses to detect object removal video forgery. In the sequential analysis, we modeled video sequences as an additive change in scalar and multidimensional parameters. The detection result at the video level is superior, but the detection result at the pixel level can be further improved. Hence, using non-additive change models may lead to better detection performance.

Furthermore, we plan to extend our approach to detect object removal forgery with a moving background. We start by segmenting a video into groups so that the camera motion within each group is smaller than a predefined threshold. Then, we perform frame registration for each group to recover the spatial alignment between the frames within the group. Subsequently, we apply sequential and patch analyses in each group.

Moreover, we plan to perform a spatiotemporal similarity test to overcome object deletion by frame-based forgery. If an object removal happens by copying a group of adjacent frames and place it on frames where the object exists, our sequential or patch analysis approach will not be able to detect a forgery because we can not detect abnormal changes within video frames. In this problem, the fundamental

challenge is to discriminate between static frames and duplicated frames. We will try to find artifacts that may occur during frames duplication to discriminate between these frames.

We also plan to design a more discriminative network architecture to detect facial manipulation. Our proposed FaceMD achieves high detection accuracy, but it can be further improved. We will investigate different network architectures using 3D convolutions. We think that 3D convolutions will improve the detection accuracy of facial manipulation because 3D convolutions extract features from adjacent video frames. Therefore, 3D convolutions would be able to expose spatiotemporal artifacts that are created during facial manipulation.

APPENDIX

We provide the copyright permissions for the articles used in this thesis. Aloraini et al., 2019 [1] and Aloraini et al., 2019 [2] are published in IS&T Electronic Imaging, Media Watermarking, Security, and Forensics 2019. Aloraini et al., 2020 [3] is published in IEEE Transactions on Circuits and Systems for Video Technology 2020.

APPENDIX (Continued)

Society for Imaging Science & Technology
7003 Kilworth Lane ▲ Springfield, VA 22151
703/642-9090; 703/642-9094 fax
info@imaging.org

July 14, 2020

Mr. Mohammed Aloraini
University of Illinois at Chicago
Department of Electrical & Computer Eng.
851 S. Morgan St.
Chicago, IL 60607

Dear Mohammed:

Thank you for your email requesting permission to use the articles below in your dissertation.

Mohammed Aloraini, Mehdi Sharifzadeh, Chirag Agarwal, and Dan Schonfeld, "Statistical Sequential Analysis for Object-based Video Forgery Detection," *IS&T Electronic Imaging, Media Watermarking, Security, and Forensics 2019*, pp.543-1—543-7, DOI: https://doi.org/10.2352/ISSN.2470-1173.2019.5.MWSF-543

Mohammed Aloraini, Lingdao Sha, Mehdi Sharifzadeh, and Dan Schonfeld, "Dictionary Learning and Sparse Coding for Digital Image Forgery Detection," IS&T*Electronic Imaging, Media Watermarking, Security, and Forensics 2019*, pp.531-1—531-6, DOI: https://doi.org/10.2352/ISSN.2470-1173.2019.5.MWSF-531

IS&T hereby grants you permission to use the article as mentioned above. Please note that the Society requires a suitable acknowledgment such as the following: "Reprinted with permission of IS&T: The Society for Imaging Science and Technology sole copyright owners of the *Electronic Imaging, Media Watermarking, Security, and Forensics 2019."*

Sincerely,

Suzanne E. Grinnan
Executive Director

SEG/dks

APPENDIX (Continued)

17 July 2020

To whom it may concern.

The IEEE does not require individuals working on a dissertation/thesis to obtain a formal reuse license however, you must follow the requirements listed below:

Textual Material
Using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © [Year of publication] IEEE.

In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appears prominently with each reprinted figure and/or table.

If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Full-Text Article
If you are using the entire IEEE copyright owned article, the following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]

Only the **accepted** version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line. You may not use the **final published** version

In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to

http://www.ieee.org/publications_standards/publications/rights/rights_link.html

to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Kind regards,

M.E. Brennan
IEEE
501 Hoes Lane
Piscataway, NJ 08854-4141

CITED LITERATURE

1. Aloraini, M., Sha, L., Sharifzadeh, M., and Schonfeld, D.: Dictionary learning and sparse coding for digital image forgery detection. IS&T Electronic Imaging: Media Watermarking, Security, and Forensics 2019, (IS&T, Springfield, VA, 2020), 2019(5):531–1–7, 2019.

2. Aloraini, M., Sharifzadeh, M., Agarwal, C., and Schonfeld, D.: Statistical sequential analysis for object-based video forgery detection. IS&T Electronic Imaging: Media Watermarking, Security, and Forensics 2019, (IS&T, Springfield, VA, 2020), 2019(5):543–1–6, 2019.

3. Aloraini, M., Sharifzadeh, M., and Schonfeld, D.: Sequential and patch analyses for object removal video forgery detection and localization. IEEE Transactions on Circuits and Systems for Video Technology (Early Access), pages 1 – 14, 2020.

4. Farid, H.: Image forgery detection. IEEE Signal processing magazine, 26(2):16–25, 2009.

5. He, Z., Lu, W., Sun, W., and Huang, J.: Digital image splicing detection based on markov features in dct and dwt domain. Pattern Recognition, 45(12):4292–4299, 2012.

6. Bayram, S., Sencar, H. T., and Memon, N.: A survey of copy-move forgery detection techniques. In IEEE Western New York Image Processing Workshop, pages 538–542. IEEE, 2008.

7. Sitara, K. and Mehtre, B. M.: Digital video tampering detection: an overview of passive techniques. Digital Investigation, 18:8–22, 2016.

8. Faceswap. `https://github.com/MarekKowalski/FaceSwap/`, Accessed: 2020-05-20.

9. Deepfakes github. `https://github.com/deepfakes/faceswap`, Accessed: 2020-05-20.

10. Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C., and Nießner, M.: Face2face: Real-time face capture and reenactment of rgb videos. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2387–2395, 2016.

11. Thies, J., Zollhöfer, M., and Nießner, M.: Deferred neural rendering: Image synthesis using neural textures. ACM Transactions on Graphics (TOG), 38(4):1–12, 2019.

12. Fan, Y., Carré, P., and Fernandez-Maloigne, C.: Image splicing detection with local illumination estimation. In 2015 IEEE international conference on Image processing (ICIP), pages 2940–2944. IEEE, 2015.

13. De Carvalho, T. J., Riess, C., Angelopoulou, E., Pedrini, H., and de Rezende Rocha, A.: Exposing digital image forgeries by illumination color classification. IEEE Transactions on Information Forensics and Security, 8(7):1182–1194, 2013.

14. Lyu, S., Pan, X., and Zhang, X.: Exposing region splicing forgeries with blind local noise estimation. International journal of computer vision, 110(2):202–221, 2014.

15. Pun, C.-M., Liu, B., and Yuan, X.-C.: Multi-scale noise estimation for image splicing forgery detection. Journal of visual communication and image representation, 38:195–206, 2016.

16. Pan, X., Zhang, X., and Lyu, S.: Exposing image forgery with blind noise estimation. In Proceedings of the thirteenth ACM multimedia workshop on Multimedia and security, pages 15–20. ACM, 2011.

17. Chennamma, H. R. and Rangarajan, L.: Image splicing detection using inherent lens radial distortion. arXiv preprint arXiv:1105.4712, 2011.

18. Hsu, Y.-F. and Chang, S.-F.: Image splicing detection using camera response function consistency and automatic segmentation. In 2007 IEEE International Conference on Multimedia and Expo, pages 28–31. IEEE, 2007.

19. Agarwal, S. and Chand, S.: Image forgery detection using multi scale entropy filter and local phase quantization. International Journal of Image, Graphics & Signal Processing, 7(10), 2015.

20. Alahmadi, A. A., Hussain, M., Aboalsamh, H., Muhammad, G., and Bebis, G.: Splicing image forgery detection based on dct and local binary pattern. In 2013 IEEE Global Conference on Signal and Information Processing, pages 253–256. IEEE, 2013.

21. Christlein, V., Riess, C., Jordan, J., Riess, C., and Angelopoulou, E.: An evaluation of popular copy-move forgery detection approaches. IEEE Transactions on Information Forensics and Security, 7(6):1841–1854, Dec 2012.

22. Fridrich, A. J., Soukal, B. D., and Lukáš, A. J.: Detection of copy-move forgery in digital images. In in Proceedings of Digital Forensic Research Workshop. Citeseer, 2003.

23. Popescu, A. C. and Farid, H.: Exposing digital forgeries by detecting duplicated image regions. Dept. Comput. Sci., Dartmouth College, Tech. Rep. TR2004-515, pages 1–11, 2004.

24. Bashar, M., Noda, K., Ohnishi, N., and Mori, K.: Exploring duplicated regions in natural images. IEEE Transactions on Image Processing, pages 1–1, 2016.

25. Ryu, S.-J., Lee, M.-J., and Lee, H.-K.: Detection of copy-rotate-move forgery using zernike moments. In Information hiding, volume 6387, pages 51–65. Springer, 2010.

26. Zandi, Mohsen Aznaveh, A. and Talebpour, A.: Iterative copy-move forgery detection based on a new interest point detector. IEEE Transactions on Information Forensics and Security, 11(11), 2016.

27. Amerini, I., Ballan, L., Caldelli, R., Bimbo, A. D., and Serra, G.: A sift-based forensic method for copy-move attack detection and transformation recovery. IEEE Transactions on Information Forensics and Security, 6(3):1099–1110, Sept 2011.

28. Huang, H., Guo, W., and Zhang, Y.: Detection of copy-move forgery in digital images using sift algorithm. In 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, volume 2, pages 272–276, Dec 2008.

29. Shivakumar, B. L., Dr, L., and Baboo, S. S.: Detection of region duplication forgery in digital images using surf. International Journal of Computer Science Issues, 2011.

30. Li, J., Li, X., Yang, B., and Sun, X.: Segmentation-based image copy-move forgery detection scheme. IEEE Transactions on Information Forensics and Security, 10(3):507–518, March 2015.

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

32. Ferreira, A., Felipussi, S. C., Alfaro, C., Fonseca, P., Vargas-Muñoz, J. E., dos Santos, J. A., and Rocha, A.: Behavior knowledge space-based fusion for copy–move forgery detection. IEEE Transactions on Image Processing, 25(10):4729–4742, 2016.

33. Pandey, R. C., Singh, S. K., and Shukla, K. K.: Passive forensics in image and video using noise features: a review. Digital Investigation, 19:1–28, 2016.

34. Mizher, M. A., Ang, M. C., Mazhar, A. A., and Mizher, M. A.: A review of video falsifying techniques and video forgery detection techniques. International Journal of Electronic Security and Digital Forensics, 9(3):191–208, 2017.

35. Johnston, P. and Elyan, E.: A review of digital video tampering: from simple editing to full synthesis. Digital Investigation, 2019.

36. Sharma, H., Kanwal, N., and Batth, R. S.: An ontology of digital video forensics: Classification, research gaps & datasets. In 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), pages 485–491. IEEE, 2019.

37. Kaur, H. and Jindal, N.: Image and video forensics: A critical survey. Wireless Personal Communications, pages 1–22, 2020.

38. Wary, A. and Neelima, A.: A review on robust video copy detection. International Journal of Multimedia Information Retrieval, pages 1–18, 2018.

39. Chao, J., Jiang, X., and Sun, T.: A novel video inter-frame forgery model detection scheme based on optical flow consistency. In The International Workshop on Digital Forensics and Watermarking 2012, pages 267–281. Springer, 2013.

40. Wu, Y., Jiang, X., Sun, T., and Wang, W.: Exposing video inter-frame forgery based on velocity field consistency. In Acoustics, speech and signal processing (ICASSP), 2014 IEEE International Conference on, pages 2674–2678. IEEE, 2014.

41. Bestagini, P., Battaglia, S., Milani, S., Tagliasacchi, M., and Tubaro, S.: Detection of temporal interpolation in video sequences. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, pages 3033–3037. IEEE, 2013.

42. Stamm, M. C., Lin, W. S., and Liu, K. R.: Temporal forensics and anti-forensics for motion compensated video. IEEE Transactions on Information Forensics and Security, 7(4):1315–1329, 2012.

43. Su, Y., Zhang, J., and Liu, J.: Exposing digital video forgery by detecting motion-compensated edge artifact. In 2009 International Conference on Computational Intelligence and Software Engineering, pages 1–4. IEEE, 2009.

44. Yang, J., Huang, T., and Su, L.: Using similarity analysis to detect frame duplication forgery in videos. Multimedia Tools and Applications, 75(4):1793–1811, 2016.

45. Wang, W. and Farid, H.: Exposing digital forgeries in video by detecting duplication. In Proceedings of the 9th workshop on Multimedia & security, pages 35–42. ACM, 2007.

46. Singh, V. K., Pant, P., and Tripathi, R. C.: Detection of frame duplication type of forgery in digital video using sub-block based features. In International Conference on Digital Forensics and Cyber Crime, pages 29–38. Springer, 2015.

47. Zhang, Z., Hou, J., Ma, Q., and Li, Z.: Efficient video frame insertion and deletion detection based on inconsistency of correlations between local binary pattern coded frames. Security and Communication networks, 8(2):311–320, 2015.

48. Wang, W. and Farid, H.: Exposing digital forgeries in video by detecting double quantization. In Proceedings of the 11th ACM workshop on Multimedia and security, pages 39–48. ACM, 2009.

49. Liao, D., Yang, R., Liu, H., Li, J., and Huang, J.: Double h. 264/avc compression detection using quantized nonzero ac coefficients. In Media Watermarking, Security, and Forensics III, volume 7880, page 78800Q. International Society for Optics and Photonics, 2011.

50. He, P., Jiang, X., Sun, T., and Wang, S.: Detection of double compression in mpeg-4 videos based on block artifact measurement. Neurocomputing, 228:84–96, 2017.

51. Sun, T., Wang, W., and Jiang, X.: Exposing video forgeries by detecting mpeg double compression. In 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 1389–1392. IEEE, 2012.

52. Ravi, H., Subramanyam, A., Gupta, G., and Kumar, B. A.: Compression noise based video forgery detection. In 2014 IEEE International Conference on Image Processing (ICIP), pages 5352–5356. IEEE, 2014.

53. Sowmya, K. and Chennamma, H.: A survey on video forgery detection. International Journal of Computer Engineering and Applications, 9(2):17–27, 2015.

54. Bagiwa, M. A., Wahab, A. W. A., Idris, M. Y. I., Khan, S., and Choo, K.-K. R.: Chroma key background detection for digital video using statistical correlation of blurring artifact. Digital Investigation, 19:29–43, 2016.

55. Su, Y., Han, Y., and Zhang, C.: Detection of blue screen based on edge features. In Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International, volume 2, pages 469–472. IEEE, 2011.

56. Xu, J., Yu, Y., Su, Y., Dong, B., and You, X.: Detection of blue screen special effects in videos. Physics Procedia, 33:1316–1322, 2012.

57. D'Amiano, L., Cozzolino, D., Poggi, G., and Verdoliva, L.: A patchmatch-based dense-field algorithm for video copy-move detection and localization. IEEE Transactions on Circuits and Systems for Video Technology, 2018.

58. Liu, Y., Huang, T., and Liu, Y.: A novel video forgery detection algorithm for blue screen compositing based on 3-stage foreground analysis and tracking. Multimedia Tools and Applications, 77(6):7405–7427, 2018.

59. Conotter, V., O'Brien, J. F., and Farid, H.: Exposing digital forgeries in ballistic motion. IEEE Transactions on Information Forensics and Security, 7(1):283–296, 2012.

60. D'Avino, D., Cozzolino, D., Poggi, G., and Verdoliva, L.: Autoencoder with recurrent neural networks for video forgery detection. Electronic Imaging, 2017(7):92–99, 2017.

61. Newson, A., Almansa, A., Fradet, M., Gousseau, Y., and Pérez, P.: Video inpainting of complex scenes. SIAM Journal on Imaging Sciences, 7(4):1993–2019, 2014.

62. Ebdelli, M., Le Meur, O., and Guillemot, C.: Video inpainting with short-term windows: application to object removal and error concealment. IEEE Transactions on Image Processing, 24(10):3034–3047, 2015.

63. Xu, R., Li, X., Zhou, B., and Change Loy, C.: Deep flow-guided video inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3723–3732, 2019.

64. Zhang, J., Su, Y., and Zhang, M.: Exposing digital video forgery by ghost shadow artifact. In Proceedings of the First ACM workshop on Multimedia in forensics, pages 49–54. ACM, 2009.

65. Hsu, C.-C., Hung, T.-Y., Lin, C.-W., and Hsu, C.-T.: Video forgery detection using correlation of noise residue. In 2008 IEEE 10th workshop on multimedia signal processing, pages 170–174. IEEE, 2008.

66. Saxena, S., Subramanyam, A., and Ravi, H.: Video inpainting detection and localization using inconsistencies in optical flow. In 2016 IEEE Region 10 Conference (TENCON), pages 1361–1365. IEEE, 2016.

67. Bagiwa, M. A., Wahab, A. W. A., Idris, M. Y. I., and Khan, S.: Digital video inpainting detection using correlation of hessian matrix. Malaysian Journal of Computer Science, 29(3):179–195, 2016.

68. Yao, Y., Cheng, Y., and Li, X.: Video objects removal forgery detection and localization. In 2016 Nicograph International (NicoInt), pages 137–137. IEEE, 2016.

69. Richao, C., Gaobo, Y., and Ningbo, Z.: Detection of object-based manipulation by the statistical features of object contour. Forensic science international, 236:164–169, 2014.

70. Chen, S., Tan, S., Li, B., and Huang, J.: Automatic detection of object-based forgery in advanced video. IEEE Transactions on Circuits and Systems for Video Technology, 26(11):2138–2151, 2016.

71. Su, L., Huang, T., and Yang, J.: A video forgery detection algorithm based on compressive sensing. Multimedia Tools and Applications, 74(17):6641–6656, 2015.

72. Lin, C.-S. and Tsay, J.-J.: A passive approach for effective detection and localization of region-level video forgery with spatio-temporal coherence analysis. Digital Investigation, 11(2):120–140, 2014.

73. Wang, X., Wang, K., and Lian, S.: A survey on face data augmentation. arXiv preprint arXiv:1904.11685, 2019.

74. Scherhag, U., Rathgeb, C., Merkle, J., Breithaupt, R., and Busch, C.: Face recognition systems under morphing attacks: A survey. IEEE Access, 7:23012–23026, 2019.

75. Zollhöfer, M., Thies, J., Garrido, P., Bradley, D., Beeler, T., Pérez, P., Stamminger, M., Nießner, M., and Theobalt, C.: State of the art on monocular 3d face reconstruction, tracking, and applications. In Computer Graphics Forum, volume 37, pages 523–550. Wiley Online Library, 2018.

76. Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., and Nießner, M.: Faceforensics++: Learning to detect manipulated facial images. In Proceedings of the IEEE International Conference on Computer Vision, pages 1–11, 2019.

77. Dale, K., Sunkavalli, K., Johnson, M. K., Vlasic, D., Matusik, W., and Pfister, H.: Video face replacement. In Proceedings of the 2011 SIGGRAPH Asia Conference, pages 1–10, 2011.

78. Garrido, P., Valgaerts, L., Rehmsen, O., Thormahlen, T., Perez, P., and Theobalt, C.: Automatic face reenactment. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4217–4224, 2014.

79. Thies, J., Zollhöfer, M., Nießner, M., Valgaerts, L., Stamminger, M., and Theobalt, C.: Real-time expression transfer for facial reenactment. ACM Trans. Graph., 34(6):183–1, 2015.

80. Doublicat. `https://reface.ai`, Accessed: 20200-05-20.

81. Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Nießner, M., Pérez, P., Richardt, C., Zollhöfer, M., and Theobalt, C.: Deep video portraits. ACM Transactions on Graphics (TOG), 37(4):1–14, 2018.

82. Agarwal, S., Farid, H., Gu, Y., He, M., Nagano, K., and Li, H.: Protecting world leaders against deep fakes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 38–45, 2019.

83. Yang, X., Li, Y., and Lyu, S.: Exposing deep fakes using inconsistent head poses. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 8261–8265. IEEE, 2019.

84. Dang-Nguyen, D.-T., Boato, G., and De Natale, F. G.: Identify computer generated characters by analysing facial expressions variation. In 2012 IEEE International Workshop on Information Forensics and Security (WIFS), pages 252–257. IEEE, 2012.

85. Carvalho, T., Faria, F. A., Pedrini, H., Torres, R. d. S., and Rocha, A.: Illuminant-based transformed spaces for image forensics. IEEE transactions on information forensics and security, 11(4):720–733, 2015.

86. Neves, J., Tolosana, R., Vera-Rodriguez, R., Lopes, V., Proença, H., and Fierrez, J.: Ganprintr: Improved fakes and evaluation of the state-of-the-art in face manipulation detection. arXiv preprint arXiv:1911.05351, 2019.

87. Matern, F., Riess, C., and Stamminger, M.: Exploiting visual artifacts to expose deepfakes and face manipulations. In 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW), pages 83–92. IEEE, 2019.

88. Li, Y., Chang, M.-C., and Lyu, S.: In ictu oculi: Exposing ai created fake videos by detecting eye blinking. In 2018 IEEE International Workshop on Information Forensics and Security (WIFS), pages 1–7. IEEE, 2018.

89. Afchar, D., Nozick, V., Yamagishi, J., and Echizen, I.: Mesonet: a compact facial video forgery detection network. In 2018 IEEE International Workshop on Information Forensics and Security (WIFS), pages 1–7. IEEE, 2018.

90. Güera, D. and Delp, E. J.: Deepfake video detection using recurrent neural networks. In 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pages 1–6. IEEE, 2018.

91. Zhou, P., Han, X., Morariu, V. I., and Davis, L. S.: Two-stream neural networks for tampered face detection. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1831–1839. IEEE, 2017.

92. Raghavendra, R., Raja, K. B., Venkatesh, S., and Busch, C.: Transferable deep-cnn features for detecting digital and print-scanned morphed face images. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1822–1830. IEEE, 2017.

93. Khodabakhsh, A., Ramachandra, R., Raja, K., Wasnik, P., and Busch, C.: Fake face detection methods: Can they be generalized? In 2018 International Conference of the Biometrics Special Interest Group (BIOSIG), pages 1–6. IEEE, 2018.

94. Tolosana, R., Vera-Rodriguez, R., Fierrez, J., Morales, A., and Ortega-Garcia, J.: Deepfakes and beyond: A survey of face manipulation and fake detection. arXiv preprint arXiv:2001.00179, 2020.

95. Li, Y. and Lyu, S.: Exposing deepfake videos by detecting face warping artifacts. arXiv preprint arXiv:1811.00656, 2018.

96. Stehouwer, J., Dang, H., Liu, F., Liu, X., and Jain, A.: On the detection of digital face manipulation. arXiv preprint arXiv:1910.01717, 2019.

97. Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-first AAAI conference on artificial intelligence, 2017.

98. Lowe, D. G.: Object recognition from local scale-invariant features, 1999.

99. Aharon, M., Elad, M., and Bruckstein, A.: K-svd: Design of dictionaries for sparse representation. In IN: PROCEEDINGS OF SPARS'05, 2005.

100. Hastie, T., Tibshirani, R., and Friedman, J.: The elements of statistical learning – data mining, inference, and prediction, 2003.

101. Fischler, M. A. and Bolles, R. C.: Random sample consensus. Commun. ACM, 24(6):381–395, June 1981.

102. Cai, T. T. and Wang, L.: Orthogonal matching pursuit for sparse signal recovery with noise. Information Theory, IEEE Transactions on, page 4688, 2011.

103. Ke, Y. and Sukthankar, R.: Pca-sift: A more distinctive representation for local image descriptors. In Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, volume 2, pages II–II. IEEE, 2004.

104. et al., M. A.: Source code of sparse representation approach and experimental results, May 2018.

105. Veth, M. J.: Fusion of imaging and inertial sensors for navigation. Technical report, Air Force Institute of Technology School of Engineering and Management, 2006.

106. Hecht, E.: Optics. Addison-Wesley, 2002.

107. Lingg, A. J., Zelnio, E., Garber, F., and Rigling, B. D.: A sequential framework for image change detection. IEEE Transactions on image processing, 23(5):2405–2413, 2014.

108. Lelescu, D. and Schonfeld, D.: Statistical sequential analysis for real-time video scene change detection on compressed multimedia bitstream. IEEE Transactions on Multimedia, 5(1):106–117, 2003.

109. Burt, P. J. and Adelson, E. H.: The laplacian pyramid as a compact image code. IEEE Transactions on Communications, 3(4), 1983.

110. Basseville, M., Nikiforov, I. V., et al.: Detection of abrupt changes: theory and application, volume 104. Prentice Hall Englewood Cliffs, 1993.

111. Bai, J.: Estimating multiple breaks one at a time. Econometric theory, 13(3):315–352, 1997.

112. Eskin, E.: Anomaly detection over noisy data using learned probability distributions. In In Proceedings of the International Conference on Machine Learning. Citeseer, 2000.

113. Tan, P.-N.: Introduction to data mining. Pearson Education India, 2018.

114. Qadir, G., Yahaya, S., and Ho, A. T.: Surrey university library for forensic analysis (sulfa) of video content. IET Conference on Image Processing (IPR 2012), 2012.

115. Lowe, D. G.: Object recognition from local scale-invariant features. In Computer vision, 1999. The proceedings of the seventh IEEE international conference on, volume 2, pages 1150–1157. Ieee, 1999.

116. Bay, H., Tuytelaars, T., and Van Gool, L.: Surf: Speeded up robust features. In European conference on computer vision, pages 404–417. Springer, 2006.

117. Alippi, C., Boracchi, G., Carrera, D., and Roveri, M.: Change detection in multivariate datastreams: Likelihood and detectability loss. arXiv preprint arXiv:1510.04850, 2015.

118. Jolliffe, I. T.: Principal Component Analysis. Springer, 2002.

119. Patwardhan, K. A., Sapiro, G., and Bertalmío, M.: Video inpainting under constrained camera motion. IEEE Transactions on Image Processing, 16(2):545–553, 2007.

120. Criminisi, A., Pérez, P., and Toyama, K.: Region filling and object removal by exemplar-based image inpainting. IEEE Transactions on image processing, 13(9):1200–1212, 2004.

121. Google-jigsaw. `https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html`, Accessed: 2020-05-20.

122. Wu, M., Trappe, W., Wang, Z. J., and Liu, K. R.: Collusion-resistant fingerprinting for multimedia. IEEE Signal Processing Magazine, 21(2):15–27, 2004.

123. Su, K., Kundur, D., and Hatzinakos, D.: Statistical invisibility for collusion-resistant digital video watermarking. IEEE Transactions on Multimedia, 7(1):43–51, 2005.

124. Danielsson, P.-E. and Seger, O.: Generalized and separable sobel operators. In Machine vision for three-dimensional scenes, pages 347–379. Elsevier, 1990.

125. Carreira, J. and Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6299–6308, 2017.

126. Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE international conference on computer vision, pages 4489–4497, 2015.

127. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1251–1258, 2017.

128. Feichtenhofer, C., Pinz, A., and Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1933–1941, 2016.

129. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467, 2016.

VITA

Name Mohammed Aloraini

Education

Ph.D., Electrical and Computer Engineering
University of Illinois at Chicago, Chicago, Illinois, United States, 2020

M.Sc., Electrical and Computer Engineering
University of Illinois at Chicago, Chicago, Illinois, United States, 2014

B.Sc., Electrical Engineering
Qassim University, Qassim, Saudi Arabia, 2011

Publications

- **M. Aloraini**, and D. Schonfeld, FaceMD: Convolutional Neural Network-Based Spatiotemporal Fusion Facial Manipulation Detection. submitted to IEEE Transactions on Circuits and Systems for Video Technology, 2020.
- **M. Aloraini**, M. Sharifzadeh, and D. Schonfeld, Sequential and patch analyses for object removal video forgery detection and localization. IEEE Transactions on Circuits and Systems for Video Technology (Early Access), pages 1 – 14, 2020.
- M.Sharifzadeh, **M. Aloraini**, and D. Schonfeld, "Adaptive Batch Size Image Merging Steganography and Quantized Gaussian Image Steganography", IEEE Transactions on Information Forensics and Security (TIFS 2019).
- **M. Aloraini**, L. Sha, M.Sharifzadeh, and D. Schonfeld, "Dictionary Learning and Sparse Coding for Digital Image Forgery Detection," IS&TElectronic Imaging, Media Watermarking, Security, and Forensics 2019, pp.531-1—531-6.
- **M. Aloraini**, M. Sharifzadeh,C. Agarwal, and D. Schonfeld, "Statistical Sequential Analysis for Object-based Video Forgery Detection," IS&T Electronic Imaging, Media Watermarking, Security, and Forensics 2019, pp.543-1—543-7.
- M. Sharifzadeh, **M. Aloraini**, and D. Schonfeld, Quantized Gaussian Embedding Steganography, IEEE International Conference on Acoustics, Speech and Signal (ICASSP 2019)
- M. Sharifzadeh, C. Agarwal, **M. Aloraini**, and D. Schonfeld, Convolutional Neural Network Steganalysiss Application to Steganography, Visual Communications and Image Processing (VCIP 2017).

Experiences

- **Teaching Assistant** at University of Illinois at Chicago, Chicago, Illinois, United States, 2016-Present.
- **Teaching Assistant** at Qassim University, Qassim, Saudi Arabia, 2011-2012.

- **Communication Engineer Co Op** at Saudi Telecom Company, Riyadh, Saudi Arabia, 2010.

Scholarships

- **Scholarship for Doctorate Degree:** Faculty scholarship program, Ministry of Higher Education, Saudi Arabia, 2015-2020.
- **Scholarship for Master Degree:** Faculty scholarship program, Ministry of Higher Education, Saudi Arabia, 2013-2014.
- **Scholarship for Diploma in English Language:** Faculty scholarship program, Ministry of Higher Education, Saudi Arabia, 2013.