Received July 10, 2019, accepted July 29, 2019, date of publication August 8, 2019, date of current version August 21, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2933871

A Novel Forgery Detection Algorithm for Video Foreground Removal

LICHAO SU, HUAN LUO, AND SHIPING WANG
College of Mathematics and Computer Science, Fuzhou University, Fujian 350108, China
Corresponding author: Huan Luo (651424071@qq.com)

ABSTRACT Video processing software is often used to remove specific moving foreground from a video. Existing forgery algorithms for detecting this type of tampering generally suffer from inefficiency and are not effective for the forged videos under complex background. To address these problems, we propose a novel forgery detection algorithm for detecting video foreground removal. The algorithm first calculates the energy factor (*EF*) of each frame to identify forged frames. An adaptive parameter-based visual background extractor (AVIBE) algorithm is then designed to detect suspected regions from the forged frames determined in the first stage. After eliminating false detection by calculating the difference of *EF* between suspected regions in the forged frames and the corresponding regions in the authentic frames, the algorithm finally locates the tampering traces. The experimental results show that our proposed algorithm has higher computational efficiency and accuracy as well as better robustness than those of previous algorithms.

INDEX TERMS Video forgery, passive forensics, foreground removal, energy factor.

I. INTRODUCTION

With the rapid development of multimedia technology and user-friendly editing software (e.g., Photoshop, Premiere by Adobe, and Mokey by Imagineer Systems), manipulating videos and changing their content is becoming a trivial task. For example, one can add or delete significant information, such as an object, from a video without leaving any visible signs of such tampering. These manipulations are sometimes not innocent, involving, for example tampering videos acquired by surveillance systems and used as evidence. Consequently, there is an increasing research interest in video forensics, which is used to authenticate the veracity and integrity of videos [1]–[4].

Video forensics can be classified into two categories: active forensics and passive forensics. In active forensics, validation information used for authentication is added during the generation of a video, i.e., digital watermarks, digital signatures and fingerprinting [5]–[7]. In passive forensics, the veracity and integrity of a video are authenticated, typically without any validation information, which is more practical in real applications than active forensics [3], [8], [9].

Recent works in passive video forensics include the following studies. In [10], the authors propose a method for detecting forged regions in the video based on inconsistencies in the noise characteristics of forged regions. Other forgery detection techniques are proposed in [11] and [12]. These techniques use double quantization coefficients to detect forgeries; however, they only work within a limited range of compression rates. In [13], an effective similarity-analysis-based method for frame duplication is proposed, which has outstanding performance in terms of computational efficiency. In [14], a novel copy-move forgery detection scheme using adaptive over-segmentation is proposed, which combines block-based method with feature-point-based method and provides strong performance. A novel video region duplication detection algorithm is proposed in [15], which has satisfactory performance for detecting videos subjected to mirror operations. In [16], a fast forgery detection algorithm based on Exponential-Fourier Moments for video region duplication is proposed which has higher computational efficiency and better robustness than those of previous algorithms.

Video forgery is mainly divided into intra-frame forgery (e.g. region duplication, foreground removal, blue screen matting) and inter-frame forgery (frame replacement, frame insertion and frame deletion) [17]. Foreground removal is an important part of intra-frame forgery and an example of foreground removal is shown in Fig. 1. However, few algorithms have been reported for detecting this type of tampering. A method based on accumulated differential images is proposed in [18], which uses textural features

The associate editor coordinating the review of this manuscript and approving it for publication was P. K. Gupta.

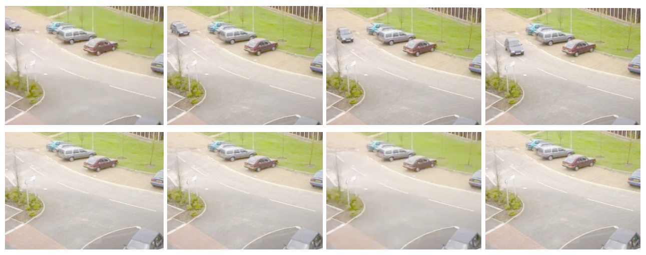


FIGURE 1. An example of video foreground removal: authentic (top) and tampered video (bottom).

around the tampered area to detect traces of moving objects removed from static backgrounds. However, the method requires the configuration of multiple empirical parameters and the detection accuracy can be affected by the shooting environment, such as trees, flowers and plants. In [19], the authors design a video forgery-detection algorithm based on spatial and temporal matching. The algorithm finds the best-matching pixels in spatial and selects the best-matching units in temporal using a variety of matching methods. However, it does not perform well for low bit-rate videos. In [20], an algorithm based on compressive sensing for detecting moving foreground removed from static background is proposed. Compressive sensing is first introduced to digital video forensics and has been found to deliver favourable performance. Nevertheless, it has a low efficiency due to complex detection procedure. An automatic algorithm in [21] is proposed to identify object-based video forgery based on a frame-manipulation detector. Experiments show that the approach achieves excellent results in both forged video identification and automatic temporal forged-segment localization. However, it is not satisfactory for videos under complex background. A fast passive forensic method to expose dynamic object removal and frames duplication using sensor noise features is proposed in [22]. The method extracts sensor noise features from each frame of video by DWT and SURE shrinkage. Then, Gaussian Mixture Density (GMD) is used as Bayesian classifier and EM algorithm set the parameters of the GMD. However, the algorithm does not perform well for videos under complex scenes.

In digital video forensics, the issues of detection accuracy and computational cost are central to the design of algorithms because even a video of modest length can run into thousands of frames. Most previous detection algorithms are frame-by-frame methods, which result in inefficiency. What's more, they are unable to detect videos under complex background (e.g., slightly shaking screens, swaying trees, water ripples, noise and brightness change). To address these problems, we propose a novel forgery detection algorithm for video foreground removal. The algorithm first calculates the energy factor (*EF*) of each frame to determine the forged frames. Then, an adaptive parameter-based visual background extractor (AVIBE) algorithm is designed to detect suspected regions in the forged frames as determined in the first stage. In order to eliminate false detection, the algorithm calculates the *EF* difference between suspected regions in the forged frames and corresponding regions in the authentic frames and finally locates the tampering traces.

The main contributions of this paper address the following elements:

1. Energy factor (*EF*) is constructed to identify forged frames, avoiding the deficiency of detecting videos frame by frame, which improves the detection efficiency greatly.
2. ViBe algorithm is first introduced into video foreground removal detection. Furthermore, we improved the ViBe algorithm with two aspects by the optimization of Euclidean distance threshold and pixel update threshold to increase the detection accuracy of the algorithm.
3. Our algorithm is effective for detecting videos under different bit rate compression. More importantly, it realizes the detection of videos under complex background (e.g., slightly shaking screens, swaying trees, water ripples, noise and brightness change), which has better robustness and is more practical in applications.

II. PRELIMINARIES

A. VISUAL BACKGROUND EXTRACTOR (VIBE) ALGORITHM

ViBe is a sample-based moving-object detection method proposed by O. Barnich that considers a moving-object detection problem as a pixel classification problem [23]. A single pixel is divided into a moving object or background pixels. ViBe uses the 1st frame to initialize the background model by building a sample set of each pixel and begins to classify new pixels from the 2nd frame. In this section, we briefly describe the principle and basic steps of the ViBe algorithm.

1) INITIALIZATION OF THE BACKGROUND SAMPLE SET

ViBe designates $v(x)$ as the value in each Euclidean colour space of the pixel located in image position x and designates v_i as a background sample value with an index i. Each background pixel x is modelled via a collection of N background sample values from previous frames, which is expressed by M.

$$M(x) = \{v_1, v_2, \ldots, v_N\} \tag{1}$$

where v_i $(\mathrm{i} = 1, \ldots, N)$ denotes the value of a sample that is randomly chosen from the 8-connected neighbourhood of each pixel according to the uniform law.

2) DETECTION OF THE FOREGROUND

To classify a pixel value $v(x)$ according to its corresponding model $M(x)$, a sphere $S_R(v(x))$ of radius R centred on $v(x)$ is first defined, as is shown in Fig. 2. Then, ViBe counts the number of samples in the background sample set contained in $S_R(v(x))$. The pixel is classified as the background if the number is larger than or equal to the given threshold, U_{min}. Otherwise, the pixel is classified as the foreground. The entire calculation method is expressed by equation (2):

$$\begin{aligned} Num &= \{S_R(v(x)) \cap M(x)\} \\ &\times \begin{cases} Num \geq U_{\min} & v(x) \in background \\ Num < U_{\min} & v(x) \in foreground \end{cases} \end{aligned} \tag{2}$$

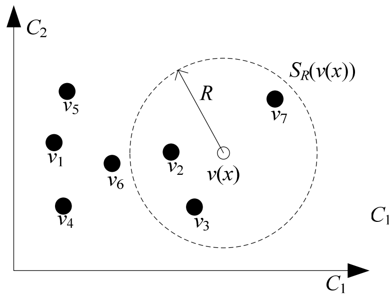


FIGURE 2. Example of a 2-D Euclidean colour space (C1, C2).

3) BACKGROUND MODEL UPDATE

In practice, the scene typically changes over time, illumination variation, the appearance of new objects or water ripples, for example. The background sample set should be updated continuously to adapt to the dynamic background.

ViBe algorithm uses a conservative background-updating algorithm. The pixels that are determined as moving targets in the current image are not included in the background update.

Considering camera vibrations, background gradients and other disruptions, ViBe algorithm uses the following method to update the background model. The pixels that are detected as moving targets in current frame don't participate in background update. When a pixel x is detected as the background, it has a $1/\varphi$ likelihood of updating a sample in the sample set $M(x)$. Simultaneously, the neighbourhood pixels also have a $1/\varphi$ likelihood of updating a sample in the sample set $M(x)$. The update process uses $v_t(x)$ ($v(x)$ at time t) to replace a sample in the sample set $M(x)$, and the updating process gradually propagates outward. The probability of a sample existing at time t_0 and still existing at time t_1 can be mathematically expressed as:

$$p(t_1 - t_0) = e^{-\ln[(n-1)/n]^{(t_1-t_0)}} \tag{3}$$

The equation above provides the probability that a sample remains in the sample set $M(x)$, which is monotone decreasing. The accuracy of background pixel estimation is improved by using the above background update method.

III. PROPOSED METHOD

Video processing software is often used to remove specific moving foreground from a video. In order to delete some moving foreground without any visual traces, the attacker may manually fill and modify the forged regions with the information provided by the areas around them. And then a gentle blur will be run over the edge of the forged regions. This tampering method requires tampering with the video frame by frame, so it is difficult for naked eyes to judge whether the video content has been tampered through colour, texture or other information of single frame. However, Due to blur operation and the large number of frames needing to be tampered in the video, the tampering traces left in the frames will cause inconsistence in temporal. More specifically, the energy ratio of high-frequency component to low-frequency component of tampered frames will be changed.

Based on this phenomenon, a novel forgery detection algorithm for video foreground removal is proposed in this paper. The algorithm consists of three stages: (1) Forged frames detection in temporal based on *EF*; (2) Suspected regions detection based on AVIBE; and (3) Tampering traces location.

The first stage of the algorithm calculates the energy factor (*EF*), which represents the proportion of low and high energy in each frame to determine the forged frames of the video. The second stage employs the AVIBE method to detect the suspected regions in the forged frames determined by the first stage. In the final stage, a series of steps is designed to eliminate detection errors and locate the tampered traces.

A. FORGED FRAMES DETECTION BASED ON EF

For a suspicious frame at a resolution of m^*n, the energy ratio of the high-frequency component to the low-frequency component is defined as follows:

$$B = \frac{1}{r}\sqrt{\sum_{i=1}^{r}\beta_i^2}\Bigg/\left(\frac{1}{m*n-r}\sqrt{\sum_{i=r+1}^{m*n}\beta_i^2}\right) \tag{4}$$

The frequency domain entropy is defined as follows:

$$H = -\sum_{i=1}^{m*n}\left(\frac{|\beta_i|}{\sum_{i=1}^{m*n}|\beta_i|}\log_2\frac{|\beta_i|}{\sum_{i=1}^{m*n}|\beta_i|}\right) \tag{5}$$

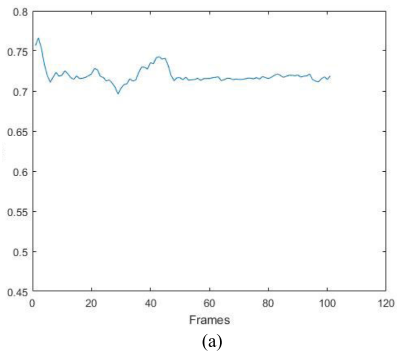
where r represents the number of low-frequency coefficients of DCT at the top left. Then the DCT coefficients are rearranged in descending order after using Z-scan technique, named new DCT coefficients, and β_i represents the i-th coefficient. The numerator in equation (4) represents the average of the low-frequency coefficients, while the denominator represents the one of the high-frequency coefficients. We have found that a larger value of B and a smaller value of H indicate more low-frequency components of a frame. Conversely, a smaller value of B and a higher value of H indicate more high-frequency components of a frame.

In order to make the change in energy of a frame more apparent, we specify *EF* to measure the degree of energy change in videos on the characteristics of B and H, which is defined as:

$$EF = \frac{1}{B} + H \tag{6}$$

Obviously, the smaller the value of *EF*, the more proportion of low-frequency information of a frame is. For an authentic video, the *EF* of each frame will keep certain continuity and consistency. Once some moving foreground has been removed from a video, filling, blurring and patching in tampering process will significantly increase the low-frequency component, and eventually result in that the *EF* of each frame in the forged frames is significantly smaller than that of frames without tampering.

Furthermore, in Fig. 3, we illustrate the *EF* curves for both the original and forged videos with *EF* on the y-axis



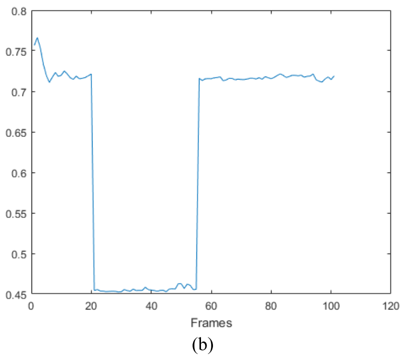


FIGURE 3. ***EF*** **for the original (a) and forged videos (b) (tampered frames: 21-55).**

and the frame number on the x-axis: (a) is the *EF* of an original video, and (b) is the corresponding forged version. We observe from Fig. 3(a) that the *EF* of each frame changes mildly and keeps certain continuity. When the video suffers from foreground removal forgery, there are obvious changes in the *EF* curve, as shown in Fig. 3(b), a sudden drop appears in the *EF* curve (forged frames: 21-55). Thus, calculating *EF* for each frame to construct the *EF* curve is a feasible method to judge whether a video has been manipulated by foreground removal. What's more, the forged frames in the video will be determined by analysing the *EF* curve.

B. SUSPECTED REGIONS DETECTION BASED ON AVIBE

Actually, removing specific moving foreground from a video is bound to leave traces in the video, even though they are invisible to the naked eye. In order to detect these traces, AVIBE algorithm is proposed in this section, which is improved on the ViBe algorithm with two aspects by the optimization of Euclidean distance threshold R and pixel update threshold φ.

1) AVIBE ALGORITHM

ViBe is a sample-based moving-object detection method with strong real-time performance. However, its global Euclidean distance threshold R and pixel update threshold φ are both fixed, which significantly impact the accuracy and robustness. When R is too small, the slow-moving objects in the background (e.g., swaying leaves or water ripples) can be easily detected as the foreground. Otherwise, they are easily regarded as the background. Meanwhile, in the ViBe algorithm, when a pixel x matches its background model, the samples of x and its neighbour pixels will be updated based on the probability of $1/\varphi$. It helps to improve the time efficiency of the algorithm, but also greatly affects the accuracy and robustness. Once the pixel belonging to the foreground is wrongly detected as the background due to noise and the background model is updated, simultaneously, it will lead to more detection errors, which greatly reduces the accuracy of the algorithm.

To overcome these limitations, firstly we introduce standard deviation $\sigma(x)$, which is used to describe the degree of changing content of each frame, and make R to be an adaptive parameter. Then, R will be adaptive adjusted according to $\sigma(x)$. The standard deviation $\sigma(x)$ is defined as

$$u(x) = \frac{1}{N}\sum_{i=1}^{N} s_i(x) \tag{7}$$

$$\sigma(x) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(s_i(x) - u(x)\right)^2} \tag{8}$$

where N represents the sample size for each pixel, $s_i(x)$ represents the i-th sample point for pixel x, and $u(x)$ represents the gray-scale average of all sample points for pixel x.

Therefore, R will be refreshed according to (9), as in:

$$R(x) = \begin{cases} R_G + \alpha * \sigma(x) & if\left(\sigma(x) \geq R_G\right) \\ R_G - \alpha * \sigma(x) & if\left(\sigma(x) \leq R_G\right) \end{cases}$$
$$R_L \leq R(x) \leq R_U \tag{9}$$

where $R(x)$ represents the adaptive Euclidean distance threshold for x pixel, while α is a constant. We note that R_G is an initial value, which is set as $R_G = 20$, R_U and R_L are the upper and lower limits of $R(x)$. Thus, the Euclidean distance threshold will be adaptively adjusted according to the video content, which improves the detection accuracy and robustness of the algorithm.

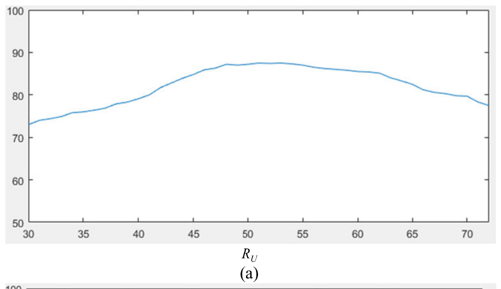
φ is the other important parameter in the ViBe algorithm, which may result in high false detection rate. In the AVIBE algorithm, the accuracy of the background model of the central pixel will be judged according to the matching information of the neighbour pixels, so as to reduce the false detection rate and improve the accuracy as well as the robustness against complex background.

Assume that NR_x represents the neighbour pixels of x with the size of $m \times m$, and q_x represents the number of pixels in NR_x that match their respective background models. The neighbourhood-matching factor $NF(x)$ is adopted to measure the correctness of background models, which is defined as

$$NF(x) = \frac{q_x}{m \times m} \tag{10}$$

According to (10), the higher value of $NF(x)$, the more pixels that match their respective background models, thus the higher accuracy of the background model. In this, $\varphi(x)$ is refreshed according to (11):

$$\varphi(x) = \frac{\varphi_G}{2 \times NF(x)}$$
$$\varphi_L \leq \varphi(x) \leq \varphi_U \tag{11}$$



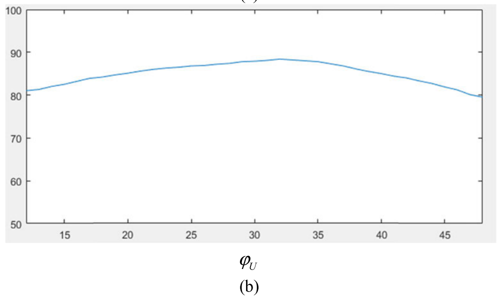


FIGURE 4. The effect of R_U *and* φ_U on the detection accuracy (*DA*).

where φ_G is an initial value, which is set as $\varphi_G = 16$, φ_U and φ_L are the upper and lower limits of $\varphi(x)$. It is indicated from (10) and (11) that the more pixels in NR_x that are consistent with the background models, the higher accuracy of the background model, thus the higher update probability.

In Fig. 4, we illustrated the effects of R_U and φ_U on the detection accuracy (DA) with "DA" on the y-axis and "R_U" (Fig. 4(a)) or "φ_U" (Fig. 4(b)) on the x-axis. The experiments were performed on 10 videos (3600 frames in total) at a resolution of 1280×720 pixels. In Fig.4(a) and (b), we fixed $R_L = 5$ and $\varphi_L = 4$ respectively. It is observed from Fig. 4 that both of two curves have tendency of ascending first and descending in succession. What's more, $R_U = 50$ and $\varphi_U = 32$ are the respective turning points of the curves. Based on this, we finally fixed the parameters as $R_U = 50$, $\varphi_U = 32$.

The screenshots of two of the sampled detection results for the AVIBE and ViBe algorithms are shown in Fig. 5. The videos using in Fig. 5 were downloaded from the Internet. (a) is the screenshots from the original videos. (b) and (c) are the screenshots of detection results for the ViBe and AVIBE algorithm, respectively. It is indicated from Fig. 5(b) that when using the ViBe algorithm, detection errors (white points in Fig. 5(b)) occur in the background, and many holes arise in the detected foreground areas. However, these problems are well solved when using the AVIBE algorithm. The moving foreground can be extracted more accurately and more completely, as shown in Fig. 5(c).

2) SUSPECTED REGIONS DETECTION

As reported in Section A, the forged frames of the video have been obtained. In this section, the AVIBE algorithm is used for detecting the suspected regions in the forged frames.

Let us consider a video sequence V, whose frames are denoted by V_t $(t = 1, 2, \ldots, L)$; the i-th frame is expressed by V_i. Assume that the forged sequence of V is from V_h to V_k.

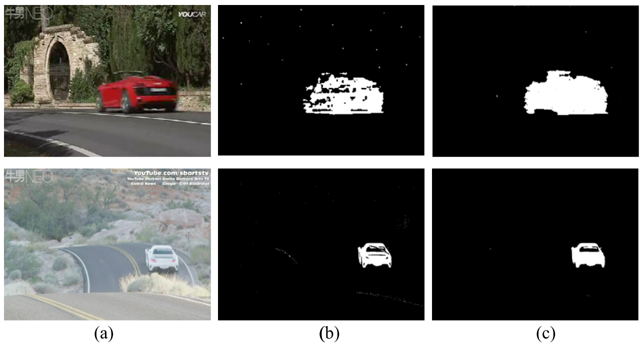


FIGURE 5. The screenshots of the detection results for the ViBe (b) and AVIBE (c).

The AVIBE is used to detect the suspected regions in V and then obtain binary images (i.e., $VB_h, VB_{h+1}, \ldots, VB_k$). Because of the few suspected regions for each frame, we construct a binary image IB_{result} to make the suspected regions more apparent, as in:

$$IB_{result} = VB_h|VB_{h+1}|\ldots|VB_{k-1}|VB_k \tag{12}$$

where | is the OR operation. In order to enhance the display effect of suspected regions, the dilation processing is applied to the IB_{result}, and then multiple suspected regions are obtained in IB_{result}, which are referred to as white connect areas. The algorithm considers that these white connect areas contain all the tampering traces in forged frames which are recorded in PS, and the i-th white area is expressed by PS_i.

C. TAMPERING TRACES LOCATION

As we know, the white connect areas obtained in previous step contain tampering traces as well as detection errors. In this section, the objective is to eliminate detection errors and locate the tampering traces by employing the *EF* algorithm, as described in Section III-A.

According to the location of the white connect areas in IB_{result}, the corresponding areas in the authentic frames are obtained. Then, the *EF* of the suspected areas in the forged frames and the corresponding areas in the authentic frames are calculated, denoted by $EF^{PS_i}_{tampered}(i = 1, 2, \ldots)$ and $EF^{PS_i}_{authenic}(i = 1, 2, \ldots)$, respectively.

If $EF^{PS_i}_{tampered} \leq cEF^{PS_i}_{authenic}$ (we fix the parameter of $c = 0.7$ based on numerous experimental analyses), then i-th white area is considered as a tampering trace, otherwise, it is eliminated. We apply this method until the last white area in IB_{result} is verified, thus locating all tampering traces in the video.

D. THE WHOLE ALGORITHM

Input the suspected video V.

Step 1: Calculate the *EF* for each frame, and construct *EF* curve to determine the forged frames of the video.

Step 2: Use the AVIBE algorithm to detect the suspected regions in the forged frames and construct a binary image. Apply dilation operation to the binary image to obtain IB_{result}.

Step 3: Calculate *EF* for all white connected areas in the forged frames and the corresponding areas in the authentic

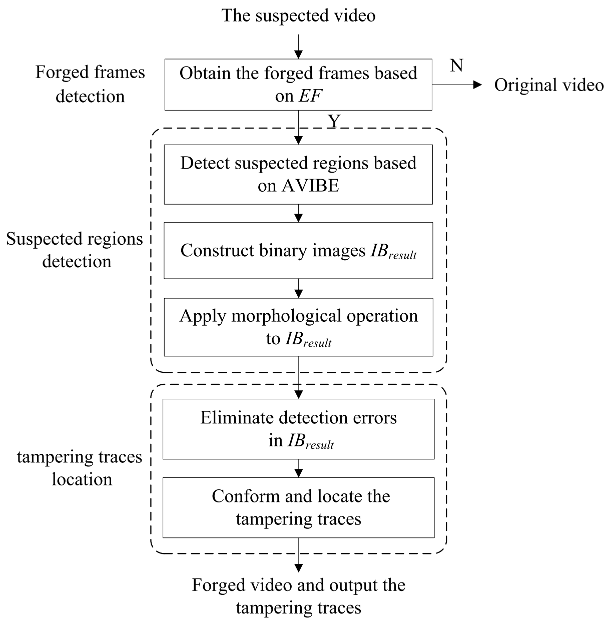

The suspected video
Forged frames detection
Obtain the forged frames based on EF
N
Original video
Y
Suspected regions detection
Detect suspected regions based on AVIBE
Construct binary images IB_result
Apply morphological operation to IB_result
tampering traces location
Eliminate detection errors in IB_result
Conform and locate the tampering traces
Forged video and output the tampering traces

FIGURE 6. The flow chart of the whole algorithm.

frames, denoted by $EF^{PS_i}_{authenic}$ and $EF^{PS_i}_{tampered}$, respectively. If $EF^{PS_i}_{tampered} \le cEF^{PS_i}_{authenic}$, then the i-th white area is considered a tampering trace, otherwise, it is eliminated.

Step 4: Mark the pixel positions of the remaining white connected areas in the video and output.

Fig. 6 shows the flow chart of the whole algorithm.

IV. EXPERIMENTS AND ANALYSIS

To evaluate the performance of the proposed algorithm, we conducted a series of experiments. The videos used in our experiments can be classified into four classes: 1) videos recorded by fixed/hand-held cameras, 2) videos downloaded from the Internet, 3) videos downloaded from the Surrey University Library for Forensic Analysis (SULFA) [24], and 4) videos downloaded from the SYSU-OBJFORG dataset (http://media-sec.szu.edu.cn/SYSU-OBJFORG.html.) (We will not public the video contents in our paper for the SYSU-OBJFORG's authors still cannot get the authorization from the people in the video scenes to use their portraits in public domain). Table 1 shows the details of some of the test videos used in the experiments.

The computer used for all the experiments in our study was configured as follows:

CPU: Intel(R) Core(TM) i7-4700 3.6 GHz.
Memory Size: 16 GB.
Video Card: NVIDIA GeForce GT 970M.
OS: Microsoft Windows 7.
Coding: MATLAB Version 7.12.0.635 (R2011a).

A. DETECTION OF FOREGROUND REMOVAL UNDER STATIC BACKGROUND

In the experiments in this section, all the videos to be used are under static background. Fig. 7 presents screenshots of some detection results.

TABLE 1. Details of the test videos.

Video	Length	Resolution	Forged frames
Video 1	140	320*240	51~140
Video 2	145	640*480	1~115
Video 3	130	640*480	1~29, 61~130
Video 4	275	320*240	1~70
Video 5	100	1280*720	20~54
Video 6	212	1280*720	121~212
Video 7	138	1280*720	1~58
Video 8	130	640*480	25~61
Video 9	210	640*480	125~210

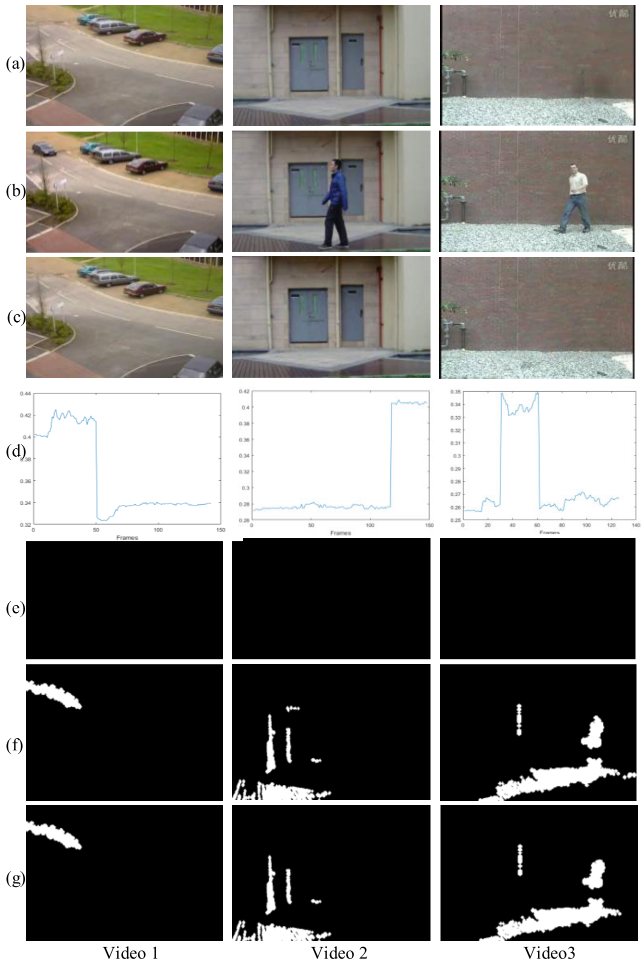


FIGURE 7. Frames and analysis of the test videos: (a) original video; (b) original video with a moving foreground; (c) tampered version of (b); (d) *EF* for (c); (e) detection results of (a) using our algorithm; (f) detection results of (c) without eliminating detection errors; (g) detection results of (c) after eliminating detection errors.

In Fig. 7, video 1 and video 3 were downloaded from the Internet, while video 2 was recorded by fixed cameras. (a) and (b) are screenshots from the original videos, where (b) contains a moving foreground. (c) is screenshots from tampered versions of (b) where the moving foreground has been removed. (d) shows the *EF* for (c). (e) shows the

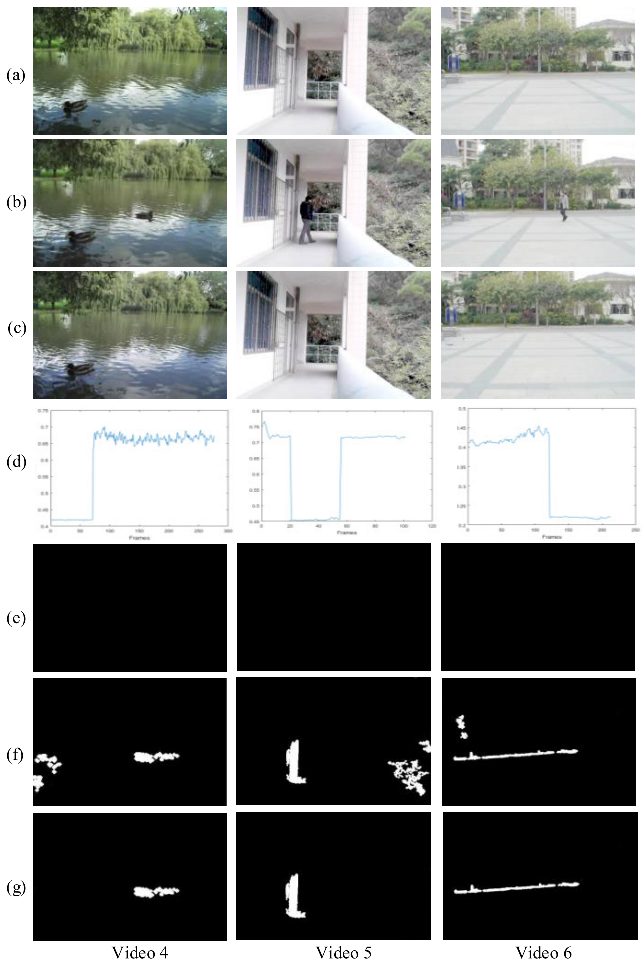


FIGURE 8. **Frames and analysis of the test videos: (a) original video; (b) original video with moving foreground; (c) tampered version of (b); (d) *EF* for (c); (e) detection results of (a) using our algorithm; (f) detection results of (c) without eliminating detected errors; (g) detection results of (c) after eliminating detection errors.**

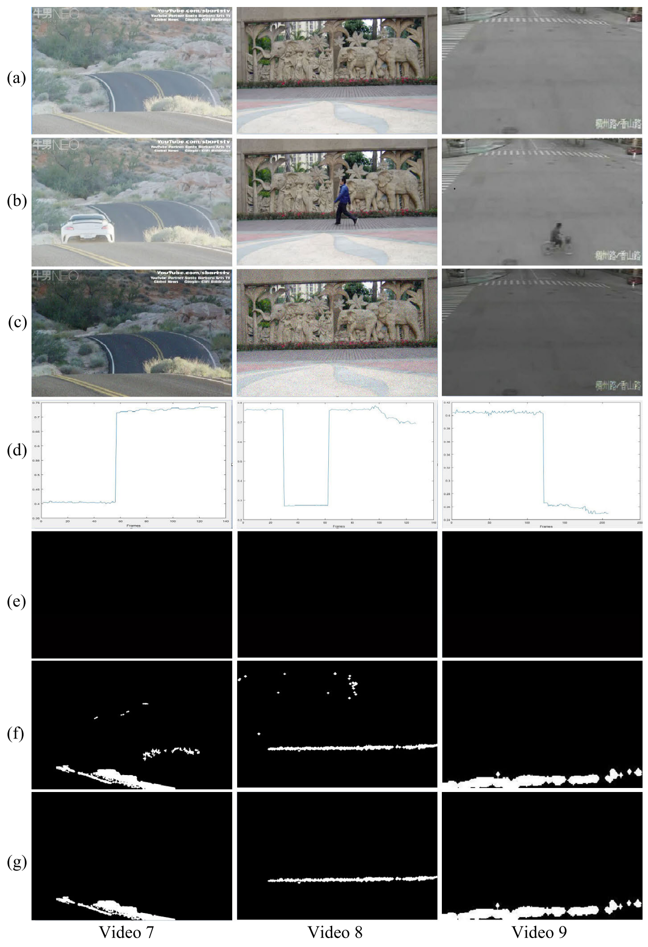


FIGURE 9. **Frames and analysis of the test videos: (a) original video; (b) original video with moving foreground; (c) tampered version of (b); (d) EF for (c); (e) detection results of (a) using our algorithm; (f) detection results of (c) without eliminating detected errors; (g) detection results of (c) after eliminating detection errors.**

detection results of (a). (f) shows the detection results of (c) without eliminating detection errors. (g) shows the final results of (c) after eliminating detection errors.

Fig. 7 shows that the proposed algorithm accurately locates the tampering traces in videos, demonstrating its effectiveness in detecting the video foreground removal forgery under static background.

B. DETECTION OF FOREGROUND REMOVAL UNDER COMPLEX BACKGROUND

In real life, the scenes of most videos typically change over time, e.g., water ripples, shaking leaves, noise and brightness change. In the experiments in this section, videos with complex backgrounds were considered. Video 4 which was downloaded from the SULFA contains water ripples. Video 5 and video 6 which were recorded by hand-held cameras with a slight shaking contain swaying trees and brush. One of the swimming ducks was removed from video 4, and the moving people in video 5 and video 6 were removed, respectively, as shown in Fig. 8 (c). Similarly, (d) shows the *EF* for (c). (e) shows the detection results of (a). (f) shows the detection results of (c) without eliminating detection errors. (g) shows the final results of (c) after eliminating detection errors. The detection results shown in Fig. 8 demonstrate the effectiveness of our algorithm in detecting the forgery of foreground removed from the video with complex background.

In Fig. 9, video 7 and video 9 were downloaded from Internet, while video 8 was recorded by hand-held cameras. After the salt & pepper noise with density of 0.01 and Gaussian white noise with variance of 0.005 were added on video 7 and video 8, respectively, the fast-moving vehicle in video 7 and the moving person in video 8 were removed. In video 9, the brightness was darkened by 40% and the cyclist was removed. In Fig. 9, (a) and (b) are screenshots from original videos. (c) is screenshots from tampered versions of (b). (d) shows the *EF* for (c). (e) shows the detection results of (a). (f) shows the detection results of (c) without eliminating detection errors. (g) shows the final results of (c) after eliminating detection errors. The detection results shown in Fig. 9 demonstrate that the proposed algorithm has better robustness to noise as well as brightness change. A major reason is

TABLE 2. Comparison of the average detection time of various algorithms for each video in different video sets.

Video	Length of each video	Resolution	ZHANG[18]	SONG[19]	SU[20]	CHEN[21]	RAMESH[22]	Proposed
Video set 1	140	320*240	2.32	4.61	18.77	30.98	10.85	**2.69**
Video set 2	275	320*240	7.95	17.13	58.95	114.58	47.61	**4.62**
Video set 3	145	640*480	10.89	28.13	96.12	160.42	81.71	**6.99**
Video set 4	130	640*480	9.79	25.67	89.18	154.33	77.02	**6.23**
Video set 5	100	1280*720	83.22	161.23	782.62	>1000s	524.36	**23.28**
Video set 6	212	1280*720	168.44	351.57	>1000s	>1000s	>1000s	**44.94**

that the Euclidean distance threshold R in AVIBE can be adaptively modified the values according to the complexity of background in video while the noise is somewhat similar to swaying grass.

C. COMPARISON OF THE EXECUTION TIME

In this section, we compared the execution time of our algorithm to those proposed in [18] (denoted as ZHANG), [19] (denoted as SONG), [20] (denoted as SU), [21] (denoted as CHEN) and [22] (denoted as RAMESH). The comparison experiments were performed on the experimental dataset of 30 videos which downloaded from SULFA and Internet and recorded by fixed/hand-held cameras, respectively. The database was divided into 6 video sets and each set contains 5 videos with the same resolution, the same frames, and the same number of forged frames but different contents. The average detection time of various algorithms for each video in different video sets are provided in Table 2.

Table 2 clearly shows that our proposed algorithm exhibits outstanding performance in terms of the computational efficiency. As is shown in Table 2, the execution time of all algorithms is closely related to the length and resolution of the video. A longer video with higher resolution indicate more execution time. Obviously from Table 2, our algorithm has higher computational efficiency than those proposed in [18] (denoted as ZHANG), [19] (denoted as SONG) and [21] (denoted as CHEN), which are frame-by-frame algorithms. Next, because of the high complexity, the efficiency of the algorithm proposed in [20] (denoted as SU) and [22] (denoted as RAMESH) are lower than that of our algorithm, although the algorithm in [17] runs every five frames.

D. COMPARISON OF THE DETECTION ACCURACY

To evaluate the capabilities of the proposed algorithm, we consider three performance indices, namely, the precision rate (PR), recall rate (RR), and detection accuracy (DA), which are defined as follows:

$$PR = \frac{TP}{TP+FP} \quad RR = \frac{TP}{TP+FN} \tag{13}$$

$$DA = \frac{TP+TN}{TP+TN+FP+FN} \tag{14}$$

TABLE 3. Performance evaluation of the proposed algorithm.

	Positive	Negative
True	93.58%	83.41%
False	6.42%	16.59%

TABLE 4. Comparison of the detection accuracy among various algorithms.

Algorithm	Detection accuracy(DA)	
	Static background	Complex background
ZHANG[18]	73.12%	32.18%
SONG[19]	80.87%	42.98%
SU[20]	86.16%	67.18%
CHEN[21]	88.79%	74.48%
RAMESH[22]	90.92%	69.18%
Proposed	**93.17%**	**86.58%**

where TP indicates that an authentic frame is detected as authentic, TN indicates that a forged frame is detected as a forgery, FP indicates that an authentic frame is detected as a forgery, and FN indicates that a forged frame is detected as authentic. Theoretically, if the algorithm used to detect foreground removal achieves higher precision and recall, its detection rate is considered better. ($TP+TN$) corresponds to the total number of detections, and ($TP+TN+FP+FN$) corresponds to the total number of frames in the experiments. Therefore, DA is the percentage of correct detection. A higher DA corresponds to a better detection rate using the proposed algorithm. Table 3 shows the performance evaluations of the proposed algorithm.

We compared the proposed algorithm with other methods in terms of accuracy. The comparison experiments were performed on the experimental dataset of 60 videos

TABLE 5. Comparison of the detection accuracy in different bit rate videos among various algorithms.

Algorithm	Detection accuracy(DA)	
	3M Bit Rate	6M Bit Rate
ZHANG[18]	39.28%	57.15%
SONG[19]	36.78%	50.35%
SU[20]	69.89%	78.51%
CHEN[21]	75.32%	83.71%
RAMESH[22]	80.55%	86.98%
Proposed	**88.12%**	**90.64%**

(28000 frames in total) downloaded from SULFA, SYSU-OBJFORG and the Internet and recorded by cameras. The experimental dataset was divided into two parts: 28 videos (13000 frames in total) with static background, and the remaining 32 videos (15000 frames in total) with complex background. The results are presented in Table 4. As is shown in Table 4, the detection accuracies of our algorithm are 93.17% and 86.58%, which are higher than those of the other methods. According to Table 4, when detecting the forgery of foreground removed from the video with static background, except for ZHANG, the detection accuracies of the algorithms are higher than 80% and only those of RAMESH and our algorithm are above 90%. By contrast, the detection accuracies of all algorithms have an appreciable decline when detecting the forgery of foreground removed from the video with complex background. Among them, the drops of ZHANG and SONG are the most because of the high background requirements of the video. Furthermore, although the methods of SU, CHEN and RAMESH show certain robustness, false detections still arise in the experiments when detecting videos with complex background.

Additionally, we compared DA of our algorithm to those of other methods in detecting different bit rate videos. The comparison experiments were performed on the videos at a resolution of 1280×720 pixels with static background downloaded from SYSU-OBJFORG and the Internet. The videos are divided into two parts: 10 videos (3600 frames) with 3M bit rate, and 10 videos (3600 frames) with 6M bit rate. The results are provided in Table 5. According to Table 5, the detection accuracies of the proposed algorithm are 88.12% and 90.64%, respectively, which are higher than those of the other algorithms. When detecting the forgery of low bit rate videos, all the algorithms decrease to some extent. Furthermore, the lower the bit rate of the videos, the lower the accuracy of the algorithms. The results in Table 5 show that videos with bit rate play the relative small role to the detection performance of our algorithm. The experimental results in Table 4 and Table 5 demonstrate that our algorithm has better robustness than those of the other algorithms.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel forgery detection algorithm that detects video foreground removal. First, the algorithm calculates EF for each frame to identify the forged frames in the video. Then, an AVIBE algorithm is used to detect the suspected regions in the forged frames as determined in the first stage. Finally, the difference of the energy factor degree between the suspected regions in the forged frames and the corresponding regions in the authentic frames is calculated to eliminate false detection and confirm the tampering traces. The experimental results show that our proposed algorithm has higher computational efficiency and accuracy as well as better robustness than those of previous algorithms.

However, it is found in the experiments that our algorithm has the following limitations:

1. The proposed algorithm is capable of detecting whether the video is subjected to tampering, but it can only locate part of the tampering traces, instead of the complete tampered regions.

2. When the foreground to be removed is too small, or the background moves too fast, the detection accuracy of the proposed algorithm will be significantly decrease.

In the future work, we will try to fix these problems and investigate to locate the tampered regions accurately. Besides of this, we will also try to extend our method for detecting other types of video forgery.

REFERENCES

[1] H. T. Sencar and N. Memon, "Overview of state-of-the-art in digital image forensics," in *Algorithms, Architectures and Information Systems Security* Singapore: World Scientific, 2009, pp. 325–347.

[2] H. Yin, W. Hui, H. Li, C. Lin, and W. Zhu, "A novel large-scale digital forensics service platform for Internet videos," *IEEE Trans. Multimedia*, vol. 14, no. 1, pp. 178–186, Feb. 2012.

[3] K. Sitara and B. M. Mehtre, "Digital video tampering detection: An overview of passive techniques," *Digit. Invest.*, vol. 18, pp. 8–22, Sep. 2016.

[4] R. D. Singh and N. Aggarwal, "Video content authentication techniques: A comprehensive survey," *Multimedia Syst.*, vol. 24, no. 11, pp. 211–240, Mar. 2018.

[5] M. Asikuzzaman and M. R. Pickering, "An overview of digital video watermarking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 9, pp. 2131–2153, Sep. 2018.

[6] X. Nie, Y. Yin, J. Sun, J. Liu, and C. Cui, "Comprehensive feature-based robust video fingerprinting using tensor model," *IEEE Trans. Multimedia*, vol. 19, no. 4, pp. 785–796, Apr. 2017.

[7] M. Fallahpour, S. Shirmohammadi, M. Semsarzadeh, and J. Zhao, "Tampering detection in compressed digital video using watermarking," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 5, pp. 1057–1072, May 2014.

[8] P. Bestagini, M. Fontani, S. Milani, and M. Barni, "An overview on video forensics," *Apsipa Trans. Signal Inf. Process.*, vol. 1, pp. 1229–1233, Dec. 2012.

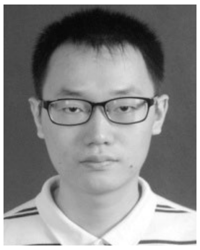
[9] O. M. Al-Qershi and B. E. Khoo, "Passive detection of copy-move forgery in digital images: State-of-the-art," *Forensic Sci. Int.*, vol. 231, nos. 1–3, pp. 284–295, Sep. 2013.

[10] M. Kobayashi, T. Okabe, and Y. Sato, "Detecting video forgeries based on noise characteristics," presented at the Pacific-Rim Symp. Image Video Technol., Tokyo, Japan, Jan. 2009.

[11] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double quantization," in *Proc. 11th ACM Workshop Multimedia Secur.*, Sep. 2009, pp. 39–48.

[12] W. Chen and Y. Q. Shi, "Detection of double MPEG compression based on first digit statistics," in *Proc. Int. Workshop Digit. Watermarking*, 2008, pp. 16–30.

[13] J. Yang, T. Huang, and L. Su, ''Using similarity analysis to detect frame duplication forgery in videos,'' *Multimedia Tools Appl.*, vol. 75, no. 4, pp. 1793–1811, Feb. 2016.
[14] C.-M. Pun, X.-C. Yuan, and X.-L. Bi, ''Image forgery detection using adaptive oversegmentation and feature point matching,'' *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 8, pp. 1705–1716, Aug. 2015.
[15] L. Su and C. Li, ''A novel passive forgery detection algorithm for video region duplication,'' *Multidimensional Syst. Signal Process.*, vol. 29, no. 3, pp. 1173–1190, Jul. 2018.
[16] L. Su, C. Li, Y. Lai, and J. Yang, ''A fast forgery detection algorithm based on exponential-Fourier moments for video region duplication,'' *IEEE Trans. Multimedia*, vol. 20, no. 4, pp. 825–840, Apr. 2018.
[17] O. I. Al-Sanjary, A. A. Ahmed, and G. Sulong, ''Development of a video tampering dataset for forensic investigation,'' *Forensic Sci. Int.*, vol. 266, pp. 565–572, Sep. 2016.
[18] J. Zhang, Y. Su, and M. Zhang, ''Exposing digital video forgery by ghost shadow artifact,'' in *Proc. 1st ACM Workshop Multimedia Forensics*, Beijing, China, Oct. 2009, pp. 49–54.
[19] Y. Song, ''Digital video forensics algorithm based on spatial and temporal matching,'' M.S. thesis, School Comput. Sci. Technol., Tianjing Univ., Tianjin, China, 2012.
[20] L. Su, T. Huang, and J. Yang, ''A video forgery detection algorithm based on compressive sensing,'' *Multimedia Tools Appl.*, vol. 74, no. 17, pp. 6641–6656, Sep. 2015.
[21] S. Chen, S. Tan, B. Li, and J. Huang, ''Automatic detection of object-based forgery in advanced video,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 11, pp. 2138–2151, Nov. 2016.
[22] R. C. Pandey, S. K. Singh, and K. K. Shukla, ''A passive forensic method for video: Exposing dynamic object removal and frame duplication in the digital video using sensor noise features,'' *J. Intell. Fuzzy Syst.*, vol. 32, no. 5, pp. 3339–3353, Apr. 2017.
[23] O. Barnich and M. V. Droogenbroeck, ''ViBe: A universal background subtraction algorithm for video sequences,'' *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1709–1724, Jun. 2011.
[24] G. Qadir, S. Yahaya, and A. T. Ho, ''Surrey university library for forensic analysis (SULFA) of video content,'' presented at the IET Conf. Image Process, 2012.



LICHAO SU received the M.E. degree in computer science and technology from Fujian Normal University, China, in 2014, and the Ph.D. degree in computer science and technology from Xiamen University, Xiamen, China, in 2018.

He is currently a Faculty Member with the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China. His current research efforts are mainly focused on video processing, multimedia forensics, and information security.

HUAN LUO received the B.Sc. degree in software engineering from Nanchang University, Nanchang, China, in 2009, and the Ph.D. degree in computer science from Xiamen University, Xiamen, China, in 2017.

He is currently a Faculty Member with the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China. His research interests include point cloud processing, computer vision, and machine learning.

SHIPING WANG received the Ph.D. degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2014. He was a Research Fellow with Nanyang Technological University, from August 2015 to August 2016. He is currently a Full Professor and the Qishan Scholar with the College of Mathematics and Computer Science, Fuzhou University. His research interests include machine learning, computer vision, and granular computing.