Information Sciences 328 (2016) 127–143

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Detecting tampering in audio-visual content using QIM watermarking

Ronaldo Rigoni[a], Pedro Garcia Freitas[a,*], Mylène C.Q. Farias[b]

[a] Department of Computer Science, University of Brasília (UnB), Brasília, Brazil
[b] Department of Electrical Engineering, University of Brasília (UnB), Brasília, Brazil

A R T I C L E I N F O

Article history:
Received 2 July 2014
Revised 17 June 2015
Accepted 23 August 2015
Available online 28 August 2015

Keywords:
Quantization Index Modulation
Watermarking
Video tampering detection

A B S T R A C T

This paper presents a framework for detecting tampered information in digital audio-visual content. The proposed framework uses a combination of temporal and spatial watermarks that do not decrease the quality of host videos. A modified version of the Quantization Index Modulation (QIM) algorithm is used to embed watermarks. The fragility of the QIM watermarking algorithm makes it possible to detect local, global, and temporal tampering attacks with pixel granularity. The technique is also able to identify the type of tampering attack. The framework is fast, robust, and accurate.

1. Introduction

Digital technologies make it possible to create high quality pictures, animations, games, and special effects with an amazing realism. Digital pictures (images and videos) can be enhanced, compressed, transmitted, converted across different formats, and displayed in a variety of devices. Also, given the significant advances in compression and transmission techniques, it is possible to deliver high quality content to the end user. As a consequence, a variety of delivery services have been created in the last few years, such as direct TV broadcast satellite, digital broadcast television, and IP-based video streaming.

This flexibility of the digital content is both a blessing and a curse. Using one of the several currently available video editing softwares, it is possible to alter (tamper) digital content without leaving any clear sign. Unauthorized users can modify and illegally distribute digital content, causing monetary and personal losses [10,12]. Therefore, a requirement for current video distribution applications is content protection, including tampering detection [22]. As a consequence, there is a great demand for automatic methods that analyze the authenticity and integrity of digital images and videos.

Several techniques have been proposed with the goal of detecting tampering in digital visual content [22]. These techniques can be divided in approaches that do not require the original (no-reference) and approaches that do require the original (full-reference). Since in most transmission applications the original is not available, the use of no-reference approaches is necessary. Unfortunately, most currently available no-reference tampering detection techniques identify a few types of tamper attacks. For example, the work of Ng et al. [16] uses higher order statistics to detect copy-and-paste (splicing) attacks, while the work of Peng and Wang detects only motion blur inconsistencies [20]. The work of Amerini et al. [1] uses a methodology based on scale invariant features transform (SIFT) to detect copy-and-paste attacks. The framework proposed by Wang et al. checks illumination angles and possible chromatic aberrations to identify traces of tampering [28]. Their method detects interpolation, double

* Corresponding author. at: University of Brasília (UnB), Department of Computer Science, Campus Universitário Darcy Ribeiro, 70910900 Brasília, Brazil, Tel.: +556186033557.
E-mail address: sawp@sawp.com.br (P.G. Freitas).

http://dx.doi.org/10.1016/j.ins.2015.08.040

interpolation, double compression, frame duplication, and re-projection (acquisition of a video from a display). These specialized algorithms have limited applications, since in practice it is difficult to ‘guess’ which type of attack was used.

Reduced-reference tampering detection represents a compromise between no-reference and full-reference approaches. Most reduced-reference algorithms embed “invisible” information (mark) into the content (host) using a watermarking technique. To verify if the original content was tampered, the algorithms extract the embedded information and verify its integrity. In the reduced-reference approach, the *fragility* of the embedded mark is a key element that determines the amount of tampering that can be detected. Among the available reduced-reference algorithms, we can cite the work of Hou et al. [35] that stores a verification bit into the 4×4 DCT blocks. Their method is robust, but it can only detect spatial tamper with a 4×4 granularity. Another example is the work of Lin et al. [13] that makes use of spatial and temporal redundancies to detect tampering in videos and images. Their method embeds a mark into high correlated blocks, what makes the algorithm more robust to geometric attacks, such as rotation and scaling.

Among the reduced-reference methods that are able to detect temporal attacks, the algorithms proposed by Subramanyam and Emmanuel [24] and by Wang and Farid [29] consider video and compression parameters (size, bitrate, and frame type) to detect temporal and spatial tampering attacks. Lin and Delp’s algorithm [13] detects compression and temporal attacks (frame switch, frame drop, and frame rate reduction). Finally, Cross et al.’s method detects MPEG-2 tampering attacks (packet losses and frame drops), but it is not able to detect the position of the tampering attack [7].

Recently, some methods were proposed to estimate video forgery using no-reference approaches. Wang et al. [30] proposed a method for video forgery detection based on the fact that forgeries introduce discontinuity points to the optical flow. Wang et al. [27] developed a no-reference method by proposing a method which assumes correlation patterns on the coefficients of greyscale values. Using local binary patterns, Zhang et al. [34] introduce an efficient method for detecting frame insertion and deletion. A novel passive video inter-frame forgery detection method was proposed by Yin et al. [32]. This method uses nonnegative tensor factorization to find some video forgeries based on the disturbs on the consistency of time-dimension factor. Although these techniques are promising, they are still restricted to the type of attack [26].

In this paper, we propose a framework for detecting and identifying tampering attacks in audio-visual content. The proposed framework is a watermark-based reduced-reference algorithm that protects both audio and video components. The algorithm detects spatial, temporal, local, and global tampering attacks. The framework is composed of a tampering protector stage and a tampering detector stage. The tampering protector generates and embeds encrypted marks into the host video and audio components. The tampering detector extracts the marks from both components, decrypts them, and compares them with the inserted marks to check if the content was tampered.

The paper is divided as follows. Section 2 presents several types of tampering attacks, which the framework can detect. Section 3 presents the tampering protection stage, while Section 4 describes the tamper detection stage. Section 5 presents the simulation results, while Section 6 presents the conclusions.

2. Tampering attacks

A digital tampering attack consists of any modification to the original digital content (image, audio, video, text, etc.). These modifications can be intentional or unintentional. Intentional tampering has the goal of maliciously modifying the content or removing the copyright. On the other hand, unintentional tampering is a consequence of digital processing operations, such as brightness correction, format conversion, size reduction, etc. For video signals, tampering attacks can be classified as spatial or temporal [5,13,14,33].

Spatial tampering attacks correspond to changes made to individual frames of the video. Spatial attacks can be further classified as local or global. Local spatial attacks correspond to changes made to a set of pixels in a frame. Examples of local attacks include changing object colors in a picture frame, removing a block of pixels, or overlapping sets of pixels. Global spatial attacks, on the other hand, modify entire video frames. Examples of global attacks include brightness adjustment, video format conversion, re-scaling dimensions, among others. Global spatial attacks can be applied to a single or to several video frames.

The most popular spatial attacks are the following:

- *Composite attack*: consists of combining two or more pictures to generate a new picture. In Fig. 1(a), a composition is made by replacing a region of the original picture frame (‘Diver’) by another image (‘UnB logo’). Notice that, if the composition is performed in a naive way, the process may generate non-homogeneous borders.
- *Cropping attack*: consists of removing parts of the picture frame. As shown in Fig. 1(b), this is a local spatial attack.
- *Flipping attack*: this is a global spatial attack that consists of rotating entire picture frames. This type of attack is difficult to detect because it keeps the watermark intact. Therefore, the detection algorithm needs to be very robust to detect the rotation of the reference watermark. An example of a Flipping attack is shown in Fig. 1(c).
- *JPEG compression attack*: consists of re-coding each picture frame with a JPEG codec. As can be seen in Fig. 1(d), this attack is almost visually imperceptible.
- *Noise addition attack*: this is a global attack that inserts noise into picture frames. Any type of noise can be used in this type of attack. Fig. 1(e) shows an example of this attack using salt-and-pepper noise.
- *Scaling attack*: consists of changing the picture frame dimensions. In the simulations performed in this work, this attack consisted of re-scaling the picture frames and discarding part of the content to make it fit into the original dimensions, as shown in Fig. 1(f).

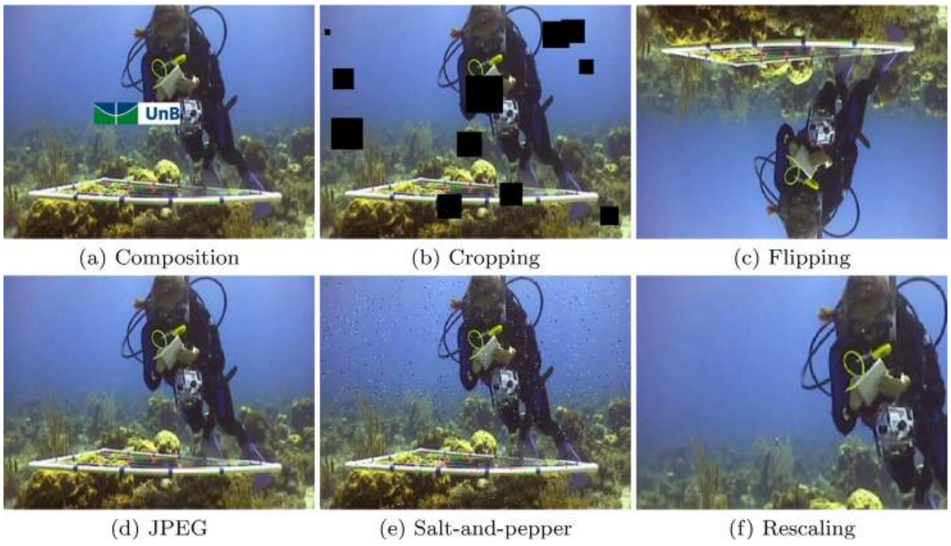

Fig. 1. Spatial tampering using the fifth frame of the 'Diver' video.

Temporal tampering attacks consist of modifying the information about the frame position (time) [12]. They include attacks such as switching the position of two (or more) frames or changing the frame rate (removing or duplicating frames). Temporal tampering attacks are hard to detect and are frequently used to confuse tampering detection algorithms. As a consequence, robustness to temporal tampering attacks may determine the resilience of the watermarking tampering detection framework [13].

The most popular temporal tampering attacks are the following:

- *Frame Duplication attack*: in this temporal attack one or more frames are copied into other temporal locations of the video. In our tests, a random number of frames are copied and pasted into a posterior position.
- *Frame Removal attack* (or FrameRate Reduction): this temporal attack reduces the number of frames of the video. This reduction is achieved by simply deleting some frames. In our tests, all videos are initially 30 frames per second (fps). The attack consists of randomly deleting one frame in a 5-frames interval.
- *Frame Switching attack*: this temporal attack consists of picking a set of random frame pairs and interchanging their positions. This attack is only visually perceptible if the interchanged frames are far away from each other. In our tests, the distance between the two frames varied from 10 to 20 frames.

In summary, in this work ten types of tampering attacks are used: Composition, Cropping, Flipping, JPEG compression, Noise Addition, Scaling, Frame Duplication, Frame Removal, and Frame Switching. Although this list is not exhaustive, the proposed framework can be extended to detect other types of attacks.

3. Tampering protection stage

The tampering protection stage generates encrypted marks and embeds them into the video and audio channels. Fig. 2 shows a block diagram of the tampering protection stage. Notice from this figure that the protection stage can be roughly divided in three steps: watermarking generation, encryption, and embedding. In this section, each of these steps is detailed.

3.1. Watermarking generation

To protect both audio and video, we first split the audio–visual content into its audio (A) and video (F) components. Then, using a pseudo-random number generator with a secret key k, we generate a mark to protect the spatial information (M_s) and another mark to protect the temporal information (M_t).

The key k is generated according to RFC 1750 Recommendations 7.1 and 7.2 [8]. The random sequence used for generating the mark complies with the random number generator tests specified in FIPS 140–2 (Security Requirements for Cryptographic Modules) [9]. More specifically, we use the Secure Hash Algorithm one (SHA-1) [17] as the basis for the Pseudo-Random Number Generation (PRNG) algorithm [3]. We compute the SHA-1 hash over a true-random value seed concatenating with a 64-bits counter which is incremented by 1 at each iteration. A 64-bits seed is used as input and we chose 64 bits out of the 160-bits of the SHA-1 output.

For each frame, M_t consists of a binary random matrix generated using the key k and replicated along the frame dimensions. Similarly, the spatial mark M_s is a random binary matrix with 5 bits of depth. Both M_t and M_s marks have the same size of host video frames and different marks are generated for each frame. The difference between the generation of spatial and temporal

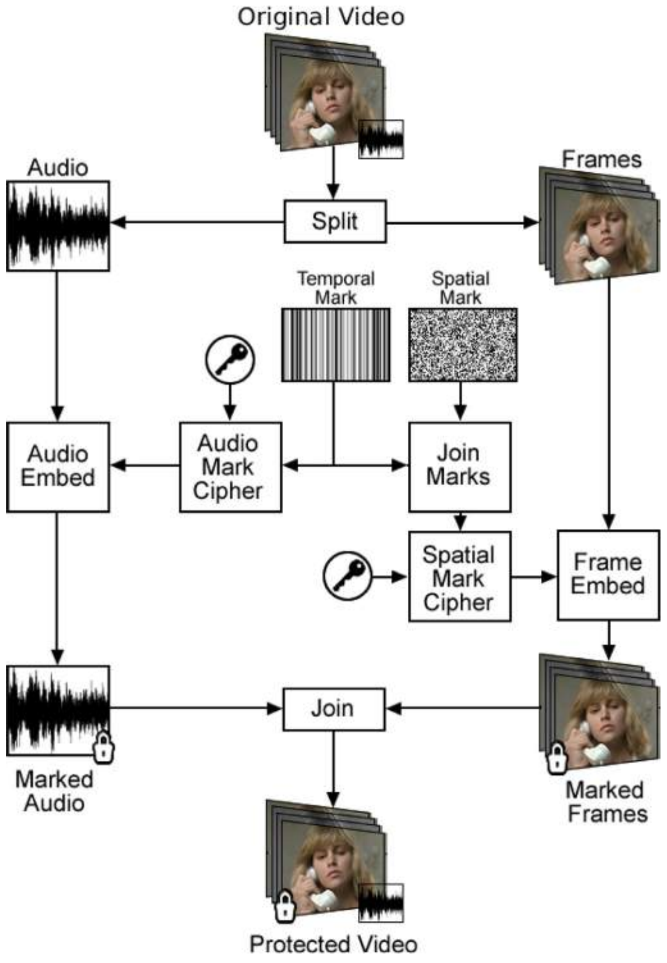

Fig. 2. Block diagram of the tampering protection stage.

marks is that for temporal marks a random number is generated for each frame. This number is replicated and embedded into the corresponding video frame.

3.2. Watermarking encryption

As mentioned earlier, both M_s and M_t are encrypted before the embedding process. The spatial mark M_s is encrypted using the equation:

$$\Psi_s[x,y] = M_s[x,y] \oplus k, \tag{1}$$

where $\oplus$ is the mathematical XOR operation, k is the encryption key, and Ψ_s is the M_s mark after encryption. Similarly, the temporal mark M_t is encrypted using the equation:

$$\Psi_t[x,y] = M_t[x,y] \oplus k, \tag{2}$$

where Ψ_t corresponds to the encrypted version of M_t.

3.3. Watermarking insertion

For the video channel F, a new mark is generated by combining Ψ_t with Ψ_s in random positions. The combination process can be described by the following equation:

$$\Phi[x,y,z] = \begin{cases} \Psi_t[x,y], & \text{if } z = i \\ \Psi_s[x,y,z], & \text{otherwise} \end{cases} \tag{3}$$

where Φ is the combined mark, $i \in [0, 5]$ is a random number generated with a pseudo-random number generator with the key k. Φ is a matrix of 6 bits of depth that contains the two marks concatenated. Since the watermark Ψ_t is also embedded in the audio component A, two copies of M_t are inserted for each frame, increasing the redundancy of M_t.

Each value of $\Phi[x, y, z]$ is embedded in the corresponding position (x, y) of the video frame. For each pixel, $\Phi[x, y, z]$ is split in three equal parts, each with 2 bits. Then, they are converted to decimal, resulting in 3 integer marks (Φ_1, Φ_2 and Φ_3). The three marks are inserted into the three color channels (RGB or YCbCr) of the video frame.

Among the available watermarking methods, the Quantization Index Modulation (QIM) algorithm is one of the methods with the best performance [4]. The QIM algorithm inserts a mark into the host signal by quantizing it with a uniform scalar quantizer. The standard quantization operation is given by the following equation:

$$Q(x, \delta) = \text{round}\left(\frac{x}{\delta}\right),$$

where *round*(.) denotes the mathematical operation of rounding a value to the nearest integer and δ is the step size used by the quantizer. The watermarked pixel is obtained using the following equation:

$$s(x) = Q(x, \delta) + d(m), \tag{4}$$

where $d(m)$ is the *perturbation value*, which depends on the mark m to be embedded.

The original QIM algorithm uses $d(m)$ as a pseudo-noise sequence. These type of embedding methods are often referred as spread spectrum methods. Some of the earliest examples of spread spectrum methods are the works by Van Schyndel et al. [25], Bender et al. [2], Cox et al. [6], and Smith et al. [23]. In fact, Bender's algorithm [2] consists of adding a small amount of δ to samples of the host signal and, then, subtracting a small amount of δ from other samples.

In this work, the mark m is randomly generated sequence of small values, which is equivalent to a pseudo-random sequence $d(m)$. Thus, the modulation function is equal to $d(m) = m$. So, the integer marks Φ_1, Φ_2, and Φ_3 are inserted into the color channels using the following equation:

$$F_m[x, y, c] = Q(F_o[x, y, c], \delta) + \Phi_c[x, y], \tag{5}$$

where F_o is the original video frame, F_m is the resulting watermarked video frame, δ is the quantization step, c is the corresponding color channel, and x and y are spatial positions. In this work $\delta = 4$, what allows embedding integer numbers with values between 0 and 3.

To embed the mark into the audio channel A, Ψ_t is reshaped into a unidimensional vector Ψ'_t. Then, Ψ'_t is embedded into A using the equation:

$$A_m[x] = Q(A_o[x], \delta) + \Psi'_t[x], \tag{6}$$

where A_m is the marked audio signal corresponding to a single video frame and A_o is the original audio channel.

After A and F are marked, the video is merged back and encoded, generating the marked video. Notice that the embedding process provides a high temporal redundancy given that several copies of the temporal mark are inserted, both in the video and audio components. Another important feature of the proposed technique is the ability to detect small changes in content. This is due to the combination of the two binary marks.

4. Tampering detection stage

The second stage of the proposed framework is the tampering detection stage, which is located at the receiver (decoder) side. Tampering detection should be performed after a transmission or storage of the video or if there is a suspicion of tampering. The tampering detection stage is roughly divided in two parts: watermarking extraction and tampering detection, as shown in the block diagram shown in Fig. 3.

4.1. Watermarking extraction

The watermarking extraction is performed on the previously marked video. First, the marks are extracted from the video and audio components. From the audio component, the mark is extracted using the following equation:

$$\hat{\Psi}'_{A,t}[x] = A_m[x] \bmod \delta, \tag{7}$$

where $\hat{\Psi}'_{A,t}$ is the extracted temporal encrypted mark, mod denotes the mathematical operation modulo, and $A_m[x]$ is the marked audio. After the extraction, the unidimensional mark $\hat{\Psi}'_{A,t}$ is reshaped into the original dimensions and becomes $\hat{\Psi}_{A,t}$. Then, $\hat{\Psi}_{A,t}$ is decrypted using the following equation:

$$\hat{M}_{A,t}[x, y] = \hat{\Psi}_{A,t}[x, y] \oplus k, \tag{8}$$

where $\hat{M}_{A,t}$ is the temporal decrypted mark extracted from A.

The marked video frames (F) contain a combination of the temporal and spatial marks. This combination is extracted, decrypted and, then, split. For each frame, the extraction of the mark is performed using the following equation:

$$\sigma[x, y, c] = F_m[x, y, c] \bmod \delta, \tag{9}$$

where $\sigma[x, y, c]$ is the extracted integer corresponding to the spatial position (x, y) at the color channel c. Notice that $\sigma[x, y, c] \in [0, 3]$.

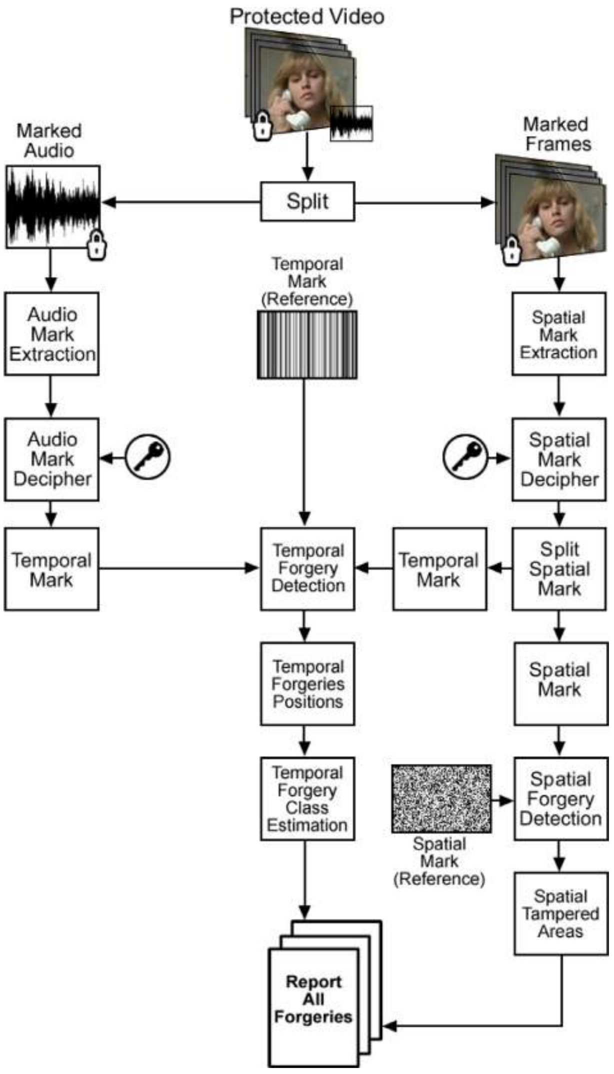

Fig. 3. Block diagram of the tampering detection stage.

Then, $\sigma[x, y, c]$ is converted to its binary representation, generating a binary matrix for each color channel: γ_r, γ_g, and γ_b. To obtain the extracted mark $\hat{\Phi}[x, y]$, these binary matrices are combined using the following equation:

$$\hat{\Phi} = \gamma_r \| \gamma_g \| \gamma_b, \tag{10}$$

where || denotes the concatenation operator. Notice that $\hat{\Phi}$ has depth equal to $3 \times \Delta$. For $\Delta = 2$, $\hat{\Phi}$ has six levels of depth. Each bit is independently addressed using the depth coordinate z.

Once $\hat{\Phi}[x, y, z]$ is created, the next step corresponds to the separation of the spatial and temporal marks. To accomplish this, a random number $i \in [0, 5]$ is generated using the same k used in the embedding process. Then, the spatial and temporal marks are extracted using the following equations:

$$\begin{cases} \hat{\Psi}_{F,t}[x, y] = \hat{\Phi}[x, y, l], & \text{if} \quad l = i \\ \hat{\Psi}_{F,s}[x, y, z] = \hat{\Phi}[x, y, l], & \text{otherwise.} \end{cases} \tag{11}$$

$\hat{\Psi}_{F,t}[x, y]$ receives the bit corresponding to the temporal mark, while $\hat{\Psi}_{F,s}$ receives the other five bits of the spatial mark. Notice that $\hat{\Psi}_{F,s}$ has a depth level of 5 bits, while $\hat{\Psi}_{F,t}[x, y]$ has a depth level of only 1 bit. At this point, both marks are encrypted and have the same size of the video frame.

The spatial mark $\hat{\Psi}_{F,s}$ is decrypted using the following equation:

$$\hat{M}_{F,s}[x, y] = \hat{\Psi}_{F,s}[x, y] \oplus k, \tag{12}$$

where $\hat{M}_{F,s}$ is the decrypted spatial mark (depth of 5 bits). Here, the same key k of the tampering protection stage is used. Likewise, the mark $\hat{\Psi}_{F,t}$ is decrypted using the equation:

$$\hat{M}_{F,t}[x,y] = \hat{\Psi}_{F,t}[x,y] \oplus k, \tag{13}$$

where $\hat{M}_{F,t}$ is the temporal decrypted mark.

After the temporal and spatial marks are extracted and decrypted, the original marks, M_s and M_t, are generated using the same key k, as described in Section 3.1. The detection process is divided in spatial detection and temporal detection, as described in next section.

4.2. Spatial detection

In spatial tampering detection, the main goal is to detect local and global attacks. For each frame, local detection is performed by comparing the extracted mark ($\hat{M}_{F,s}$) with the original mark (M_s) using the following equation:

$$\rho_{F,s}[x,y] = \begin{cases} \text{true}, & \text{if } \hat{M}_{F,s}[x,y,z] \neq M_s[x,y] \\ \text{false}, & \text{otherwise} \end{cases} \tag{14}$$

where $\rho_{F,\,s}$ is a boolean *Spatial Detection Matrix* (SDM). In this matrix, each position that has 'true' values corresponds to a 'tampered' pixel. Notice that using this approach, the exact location of a tampered area can be detected.

When the locations of the tampered areas are detected, the SDM is analyzed to find the percentage of tampered area in relation to the total area of the video. The idea here is to try to characterize the changes to the video in order to classify the type of attack. For example, if this percentage is greater than a threshold, $\tau_{F,\,s}$, this frame is classified as "globally" attacked. Otherwise, it is classified as "locally" attacked.

Notice that both local and global tamperings damage the temporal mark inserted in the video frame. This is a problem. If the temporal mark is lost, the temporal detection might fail. To avoid this problem, the temporal mark that is replicated in the audio channel is used. This allows the detector to differentiate temporal, local, or global attacks, as described next.

4.3. Temporal detection

The temporal detection algorithm uses the temporal marks embedded in the video and audio components, F and A, respectively. First, it estimates the *Temporal Detection Matrix* (TDM) using the following equation:

$$\rho_{F,t}[x,y] = \begin{cases} \text{true}, & \text{if } \hat{M}_{F,t}[x,y] \neq M_t[x,y] \\ \text{false}, & \text{otherwise} \end{cases} \tag{15}$$

where M_t is the original temporal mark (reference) and $\rho_{F,\,t}$ is a boolean matrix. When $\rho_{F,\,t}$ contains 'true' values, a watermark loss has occurred and the corresponding frame is probably temporally tampered. To detect if a frame is temporally tampered, the percentage of 'true' values in TDM is compared with a threshold $\tau_{F,\,t}$. If this percentage is less than $\tau_{F,\,s}$, this frame is classified as NOT temporally attacked.

Notice that the 'true' percentage is not a very accurate criteria to determine temporal tampering because the spatial tampering may damage the temporal mark. To double check if the frame is temporally attacked, the audio mark can be used when $\rho_{F,\,t}$ is higher than $\tau_{F,\,t}$. The mark ($\hat{M}_{A,t}$) is compared with the original mark using the following equation:

$$\rho_{A,t}[x,y] = \begin{cases} \text{true}, & \text{if } \hat{M}_{A,t}[x,y] \neq M_t[x,y] \\ \text{false}, & \text{otherwise} \end{cases} \tag{16}$$

where $\rho_{A,\,t}$ is the TDM of the audio mark. From $\rho_{A,\,t}$, the percentage of 'true' values is compared to a threshold $\tau_{A,\,t}$. When the lost percentage in $\rho_{A,\,t}$ is smaller than $\tau_{A,\,t}$, the frame is not temporally tampered. Therefore, the detection using $\rho_{F,\,t}$ indicates that spatial tampering can damage the temporal mark embedded in the frame (video component). If the criteria for audio and video comparison show that temporal watermarking loss has occurred, the frame is classified as temporally attacked.

Next, a queue Θ is created to estimate the temporal attack type. This queue stores the temporal detection result. If the frame is classified as 'temporally attacked', Θ stores a copy of the extracted temporal mark. Otherwise, Θ stores $\varnothing$ (null) in the corresponding frame position.

4.3.1. Temporal tampering position

Once the temporal attack location is determined, the attack type can be estimated. For this, the original temporal mark is generated and stored in a queue Ω. Then, Θ is analyzed using Ω as a reference.

The first analysis consists of searching all occurrences in Ω that do not occur in Θ. If mismatches are found, the distances among all tampering attacks are computed. If all occurrences have the same distance, this attack is classified as a Framerate reduction. Otherwise, it is classified as Frame Deletion attack.

The second analysis consists in verifying if all elements of Ω are also in Θ and if the index ordering is kept. Also, Θ is checked to verify if it contains elements that does not exist in Ω. If both conditions are satisfied, this means that Θ contains more elements than the original Ω. This indicates that new frames were inserted into the video and the attack is classified as a Frame Insertion attack.

The third analysis consists of searching all Ω elements that occur more than once in Θ. This means that one or more frames were copied and pasted into other video positions. So, this attack is classified as Frame Duplication attack.

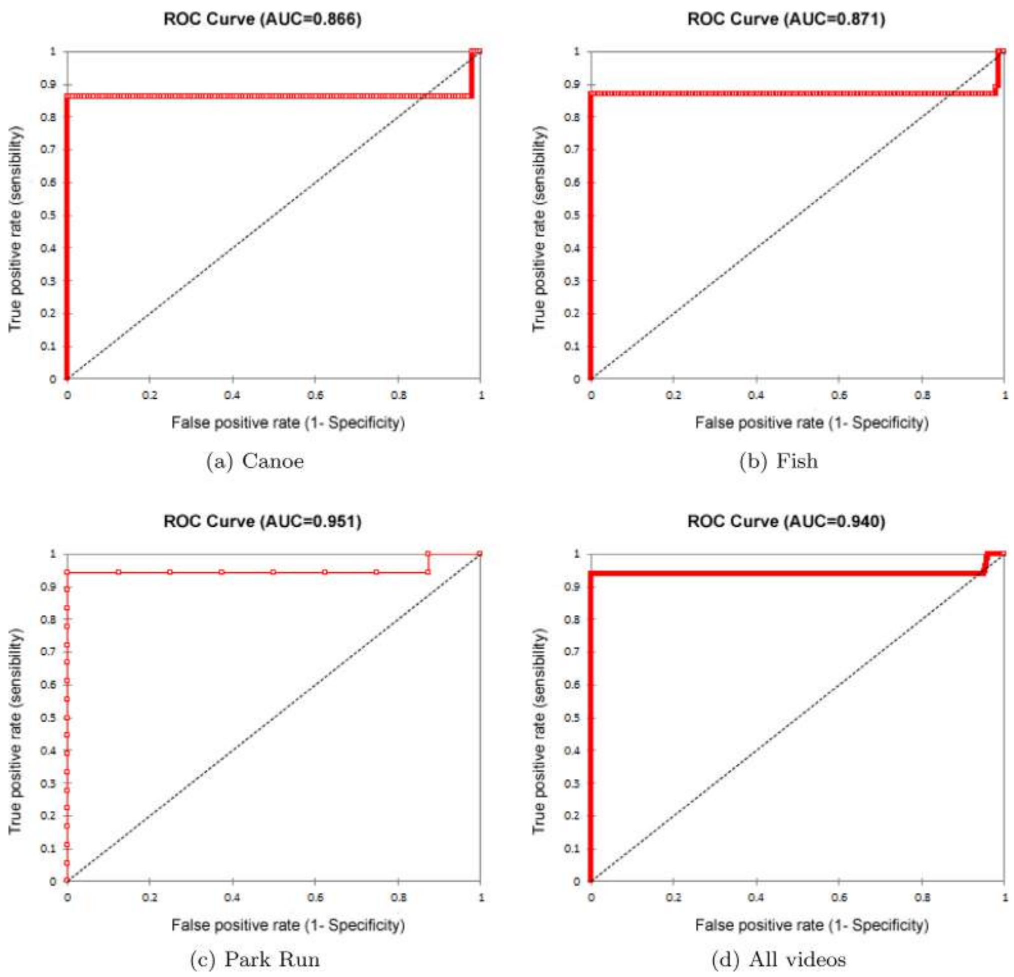

Fig. 4. Sensitivity versus specificity classification curves for (a) Canoe, (b) Fish, (c) Park Run, and (d) all videos.

Fourth, for each element marked as tampered at position $i \in \Theta$, the correct position $j \in \Omega$ is found. Then, the element $\Theta[j]$ is verified to check if it is equal to $\Omega[i]$. If they are equal, then this attack is classified as a Frame Switch attack.

4.4. Thresholds analysis

To determine the optimal value of the thresholds defined in Sections 4.2 and 4.3, a sensitivity analysis was performed. Foremost, the true positive rate was computed against the false positive rate at various threshold settings for each video. In other words, for each video, the spatial attacks were classified (local or global) and the appropriate threshold ($\tau_{F,s}$) values were computed. This analysis is directly related to a cost-benefit analysis of diagnostic decision making.

Fig. 4 shows the receiver operating characteristic (ROC) curve of the attack classification (local or global) for the videos (a) “Canoe”, (b) “Fish”, and (c) “Park Run”. In a similar way, the classification was computed for all videos, as shown in Fig. 4(d). In this figure, the classification model is a mapping of instances between global and local classes. The classifier boundary between classes must be determined by a threshold value. The best values of these thresholds are listed in Table 1. From this table, it is possible to notice that the best value for $\tau_{F,s}$ is, on an average, 85.593%.

5. Simulations and results

In this work, two sets of copyright-free videos are used. The first data set consists of videos from the ReefVid database, which is maintained by the University of Queensland, Australia [15]. These videos are in AVI format, compressed with MPEG-4 (framesize 384×288, frame rate 25 fps). The audio content in these videos were inserted artificially. Sample frame thumbnails of this set are shown in Fig. 5.

The second set of video sequences used in this work are from the Consumer Digital Video Library (CDVL) [21]. The videos have a resolution of 1280×720, color space of 4:2:0, and frame rate of 30 frames per second (fps). All videos had accompanying audio. Sample frame thumbnails are shown in Fig. 6.

Table 1
AUC, optimal thresholds and precision of analyzed videos.

Video	AUC	Best threshold	Precision
1	0.866	88.063	0.902
2	0.87	86.364	0.899
3	0.852	85.084	0.888
4	0.871	85.452	0.906
5	0.855	87.681	0.897
6	0.893	83.478	0.92
7	0.951	88.389	0.962
8	0.866	87.179	0.899
9	0.916	88.333	0.938
10	0.782	88.139	0.853
11	0.907	81.818	0.928
12	0.891	83.842	0.915
13	0.888	84.172	0.913
14	0.873	84.53	0.906
15	0.913	81.367	0.933
Average	0.879	85.593	0.911
All	0.940	81.367	0.954

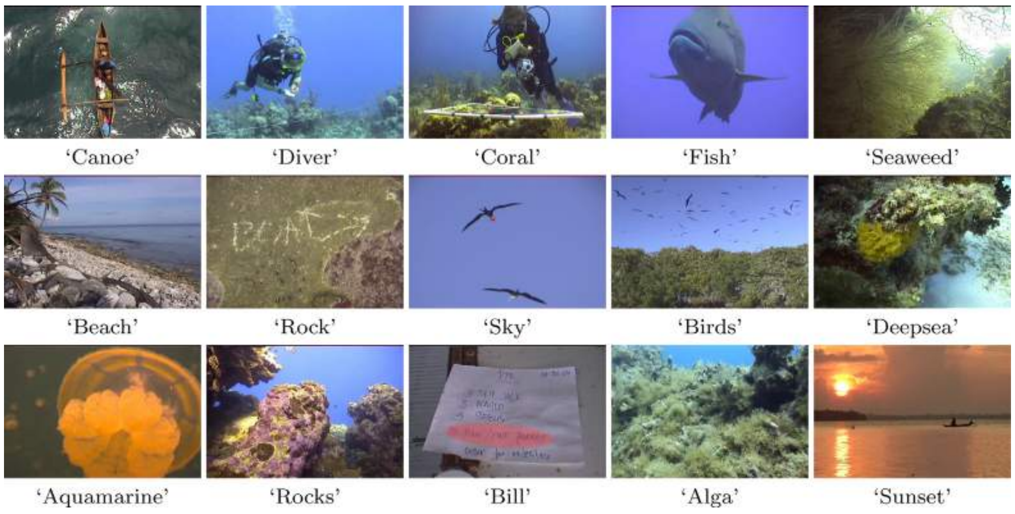

Fig. 5. Frame thumbnails of ReefVid video sequences used in the simulations.

Fig. 6. Frame thumbnails of CDVL video sequences used in the simulations. Database can be downloaded from www.cdvl.org or http://www.gpds.ene.unb.br/mylene/2013-UNB-AV/testseq/.

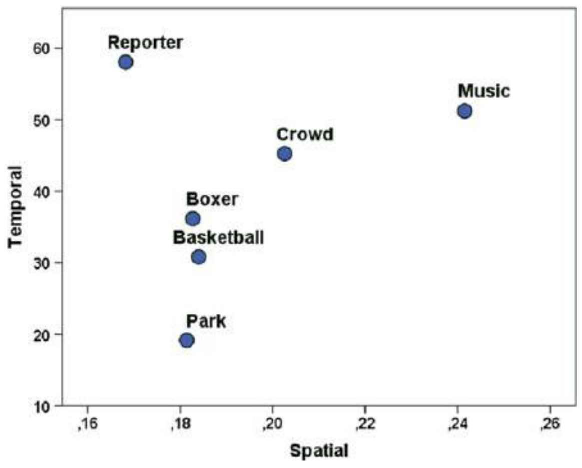

Fig. 7. Spatial and temporal perceptual information measures.

Table 2
Impact of marks on test videos for ReefVid database.

Video	Frames	PSNR	SSIM
Canoe	413	45.53459	0.99966
Diver	337	45.40895	0.99968
Coral	314	45.51629	0.99967
Fish	612	45.56291	0.99964
Seaweed	191	45.44542	0.99937
Beach	803	45.59478	0.99955
Rock	19	45.37220	0.99967
Sky	273	45.45621	0.99966
Birds	102	45.38711	0.99967
Deepsea	58	45.41717	0.99958
Aquamarine	1123	45.51627	0.99940
Rocks	751	45.57977	0.99973
Bill	900	45.54493	0.99936
Alga	576	45.55015	0.99966
Sunset	1938	45.64242	0.99970

All test sequences were chosen to guarantee good spatial and temporal distributions. Some videos have a higher spatial activity (texture), while others have higher temporal activity (movement). The audio content was also taken into account, selecting sequences that had speech, music, and ambient sound. This diversity is an important requirement when choosing a video database for testing tampering attacks.

Fig. 7 shows the spatial and temporal information measures for the CDVL video database (computed as defined by Ostaszewska and Kloda [18]). Notice that the video “Reporter” has the highest temporal activity and the lowest spatial activity. The video “Music” has both a high temporal activity and a high spatial activity, while the video “Park Run” has relatively low spatial and temporal activities.

The first test performed in this work consists of analyzing the quality of the *marked* videos. Although the goal of the proposed framework is to protect the video content from tampering, it is important that the overall quality of the videos are not decreased by the watermarking algorithm. This analysis was made by comparing the frames of marked and original videos using two popular full-reference visual quality metrics: the Peak Signal-to-Noise Ratio (PSNR) and Structural SIMilarity (SSIM) index [31]. Table 2 shows the average SSIM and PSNR values for all marked frames of the ReefVid database. The PSNR values obtained are all above 45 dB, while the SSIM values are all above 0.999. These values indicate that the marked videos have a very good quality with visually imperceptible distortions.

Next, 10 types of attacks (three temporal and seven spatial – see Section 2) were performed on the database images. For the spatial attacks, the same tests were performed for all videos of the ReefVid and CDVL databases. But, due to lack of space, only the results for the video ‘Sunset’ are shown. Fig. 8 shows the proportion of attacked (tampered) and detected regions for all frames of this video. The attacks correspond to the areas modified using several types of tampering. The darker bars are the proportions of areas detected as tampered, while the lighter bars are the proportions of the corresponding attacks. It is worth pointing out that less than 25% of the video is attacked and, therefore, the bars do not reach the 100% limit. When the bars have the same size this means that the proposed framework was able to detect all tampering attacks.

Fig. 9 shows the detection rate by frame for the video ‘Diver’. The darker bars are the proportions of areas detected as tampered, while the lighter bars are the proportions of the corresponding attacks. The proposed framework is able to detect more

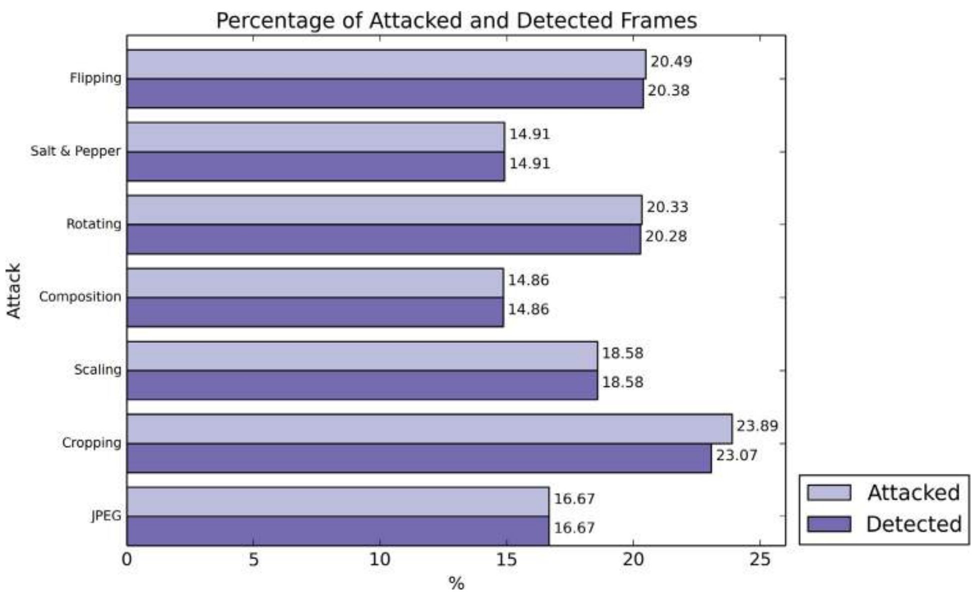

Fig. 8. Global spatial tampering detected for all frames of 'Diver' video.

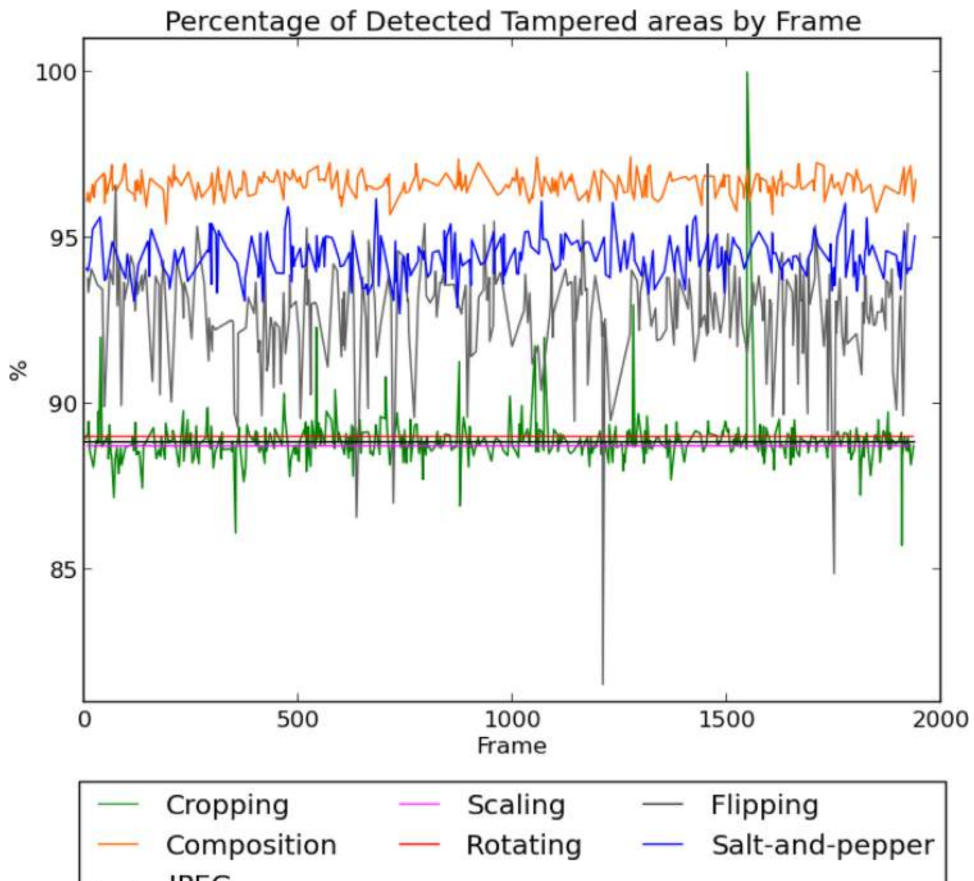

Fig. 9. Percentage of detected local attack for each frame of 'Diver' video.

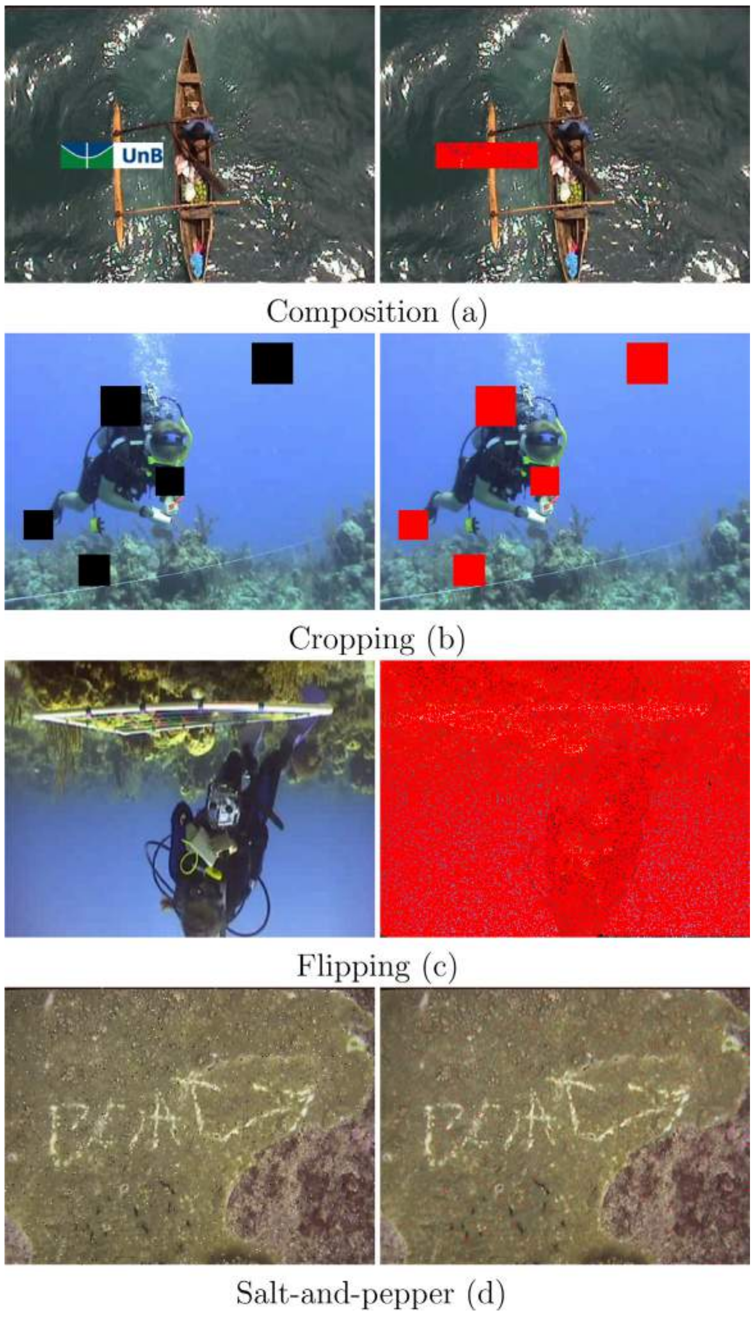

Fig. 10. Spatial attacks (left) and their detections (right) of five tested videos. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article).

than 85% of tampering attacks for all analyzed attacks.When both light and dark bars have the same size, the proposed framework is able to detect all tampering attacks. As can be noticed, the worst performance is obtained for Cropping attacks. In Cropping attacks, several attacked areas may overlap into a single region. In these cases, the framework identifies only one attacked area. When comparing Figs. 9 and 8, we notice an apparent disparity between JPEG attack detection. This is because for JPEG compression attacks, some high frequencies are not eliminated and the embedded mark is not completely lost. As a consequence, the regions are not classified as tampered (Fig. 8), but the frame can be classified as tampered (Fig. 9).

Fig. 10 shows four spatial attacks and their detections. In this figure, the red highlights illustrates the tampered regions. The first line shows Composition, Cropping, Flipping, and Salt-and-pepper (Noise) attacks. The second line shows their corresponding detections. Thus, Fig. 10(a) shows the Composition attack and its detection for a frame of the video ‘Cannoe’. Fig. 10(b) illustrates the Cropping attack and its corresponding detection for the video ‘Diver’. Fig. 10(c) shows an example of a 180 degrees Flipping

Table 3
Efficiency of spatial detections per area for ReefVid database (%).

Video	Composition	Flip	JPEG.	Rotation	Noise	Scale	Cropping
Canoe	96.57	88.89	93.65	88.86	94.37	88.84	88.88
Diver	96.53	88.79	93.52	88.85	94.41	88.85	88.65
Coral	96.66	88.95	93.20	88.86	94.38	88.85	88.74
Fish	96.62	88.90	92.99	88.86	94.22	88.83	88.81
Seaweed	96.53	88.86	93.49	88.86	94.52	88.83	88.77
Beach	96.65	88.88	93.54	88.86	94.40	88.84	88.79
Rock	96.82	88.92	93.22	88.88	94.61	88.83	86.91
Sky	96.66	88.88	92.80	88.86	94.46	88.86	88.98
Birds	96.59	88.90	93.60	88.86	94.24	88.85	88.94
Deepsea	96.65	88.87	93.66	88.86	94.40	88.80	88.55
Aquamarine	96.62	88.81	93.52	88.85	94.29	88.83	88.83
Rocks	96.62	88.81	93.48	88.86	94.34	88.79	88.68
Bill	96.68	88.81	92.89	88.85	94.37	88.86	88.71
Alga	96.54	88.88	93.49	88.87	94.37	88.86	88.87
Sunset	96.65	88.84	92.69	88.86	94.42	88.86	88.83
Average	96.63	88.87	93.32	88.86	94.39	88.84	88.66

Table 4
Efficiency of spatial detections per area for CDVL database (%).

Video	Composition	Flip	JPEG.	Rotation	Noise	Scale	Cropping
Boxer	96.66	88.93	93.35	88.88	94.43	88.88	89.01
Park Run	96.72	88.89	93.64	88.88	94.49	88.88	88.91
Crowd Run	96.78	88.89	93.56	88.87	94.45	88.88	88.95
Basketball	96.72	88.90	93.43	88.87	94.56	88.87	88.94
Music	96.64	88.90	92.92	88.87	94.48	88.88	88.89
Reporter	96.74	88.79	92.81	88.87	94.44	88.88	88.92
Average	96.71	88.88	93.28	88.87	94.47	88.88	88.94

attack and its detection for the same video. Notice that, for the Flipping attack, most of the frame is detected as being tampered since the majority of the spatial positions of the mark has been altered. Finally, Fig. 10(d) shows a Salt-and-pepper noise attack and its detection for the video 'Rock'.

Table 3 shows the average efficiency percentage of seven spatial attacks, for all tested videos of the ReefVid database. The average efficiency percentage is computed as the ratio between the amount of detected areas and the amount of attacked areas. From this table, the framework has around 96% of successful detections for the Composition attack. The average of false-negative percentages is 4%, since the image inserted by the Composition attack has always the same size. Flipping, Rotating, and Scaling attacks have an average of false-negative percentages of around 12%, since these are all global attacks in which the watermark is lost in the same way. For the JPEG compression attack, since the compression factor used by the attack is always 95%, the average efficiency percentage is around 93% for all videos. Since the Salt-and-pepper (Noise) attack is applied randomly to frames, attacked positions may overlap and detection may fail in these overlapping positions. As a result, this attack has an average false-negative percentage of 4%. A similar situation of overlapping positions is encountered for the Cropping attack, causing an average false-negative percentage of 12%.

These same tests were performed for the videos from the CDVL database. As shown in Table 4, the results are similar to those presented in Table 3. Considering that the CDVL videos have a higher spatial resolution and contain other audio/video characteristics, this implies that the prediction accuracy of the proposed method is invariant to the spatial resolution and to the audio and video characteristics.

As mentioned earlier, in this work three types of temporal attacks were performed. Test results for two types of temporal attacks are shown in Figs. 11 and 12. Fig. 11 shows a Frame Duplication attack for the video 'Coral'. In the first line of the figure, the 28-th frame was copied and pasted between the 28-th and 29-th frame. The detection is shown in the second line of the figure. Fig. 12 shows the result for a Switch attack of the video 'Sunset'. On the first line, the 163-th and 529-th frames of the video have their position interchanged. On the second line of the figure, the detection is presented. The last tested attack is the FrameRate reduction, which have similar results.

The average efficiency and accuracy of the temporal detection for the ReefVid and CDVL databases are shown in Tables 5 and 6, respectively. In Frame Switch, an overlapping of the attacked positions occurred, i.e. some frames were switched more than once and ended up back on its original position. For FrameRate and Duplication attacks, the framework is able to detect almost all tampered frames.

While Tables 5 and 6 show the average efficiency percentage for all videos, Fig. 13 shows the exact number of detected and attacked frames for the 'Sunset' video. For Frame Duplication and FrameRate reduction, there were no false-positives, i.e. the detector was able to detect 100% of all test attacks. For Frame Switch, there were only 7.8% of false-negatives for a video with 1938 frames (Fig. 14).

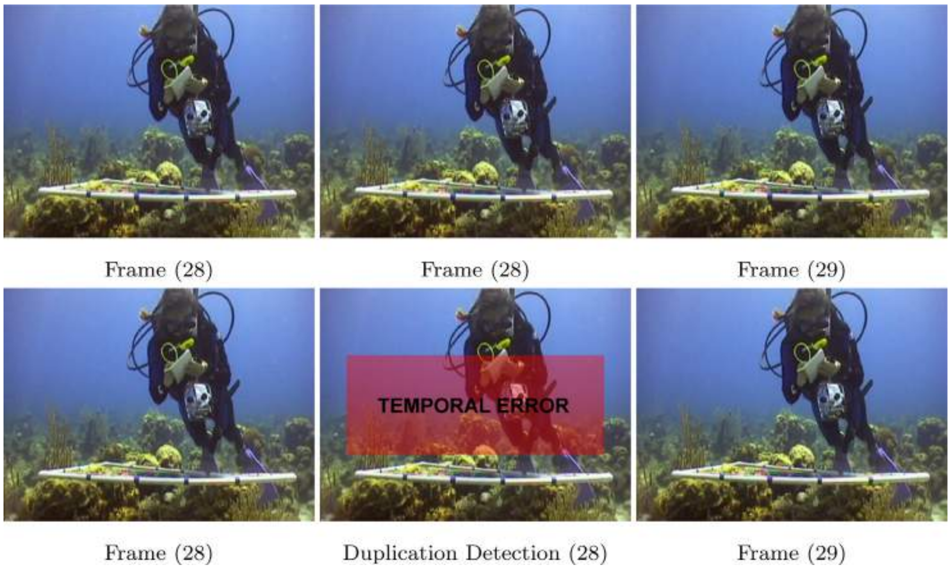

Fig. 11. Frame duplication attack and its detection for video ‘Coral’.

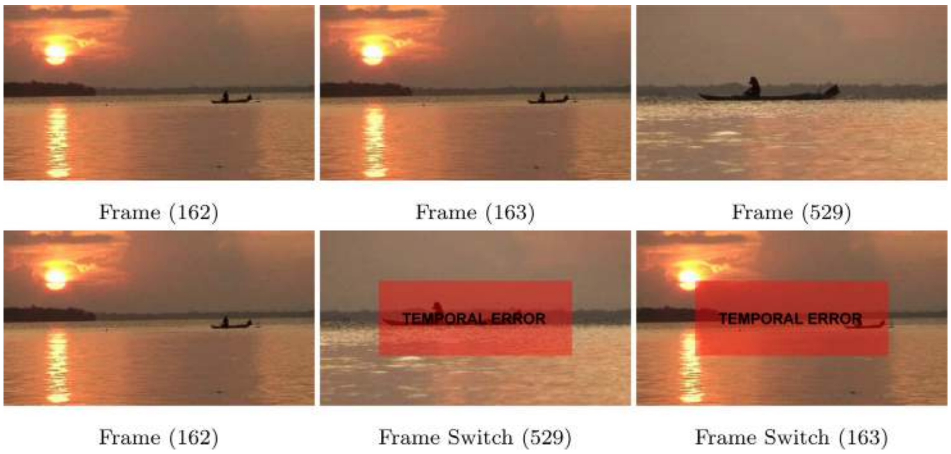

Fig. 12. Temporal Frame Switch attack for video ‘Sunset’ and its detection.

Table 5
Average percentage efficiency of temporal detections for ReefVid database (%).

Video	Switch	FrameRate	Duplication
Canoe	100.0	100.0	100.0
Diver	83.3	100.0	100.0
Coral	91.6	100.0	100.0
Fish	91.6	100.0	94.7
Seaweed	100.0	100.0	100.0
Beach	90.0	100.0	100.0
Rock	80.0	100.0	100.0
Sky	63.1	100.0	100.0
Birds	71.4	100.0	100.0
Deepsea	100.0	100.0	100.0
Aquamarine	100.0	100.0	100.0
Rocks	100.0	100.0	100.0
Bill	100.0	97.4	100.0
Alga	94.1	96.0	100.0
Sunset	92.8	100.0	100.0
Average	90.5	99.5	99.6

Table 6
Average percentage efficiency of temporal detections for CDVL database (%).

Video	Switch	FrameRate	Duplication
Boxer	83.3	100.0	100.0
Park Run	100.0	90.0	91.6
Crowd Run	89.4	100.0	100.0
Basketball	50.0	91.6	98.1
Music	92.3	90.9	87.5
Reporter	100.0	100.0	96.2
Average	85.8	95.4	95.5

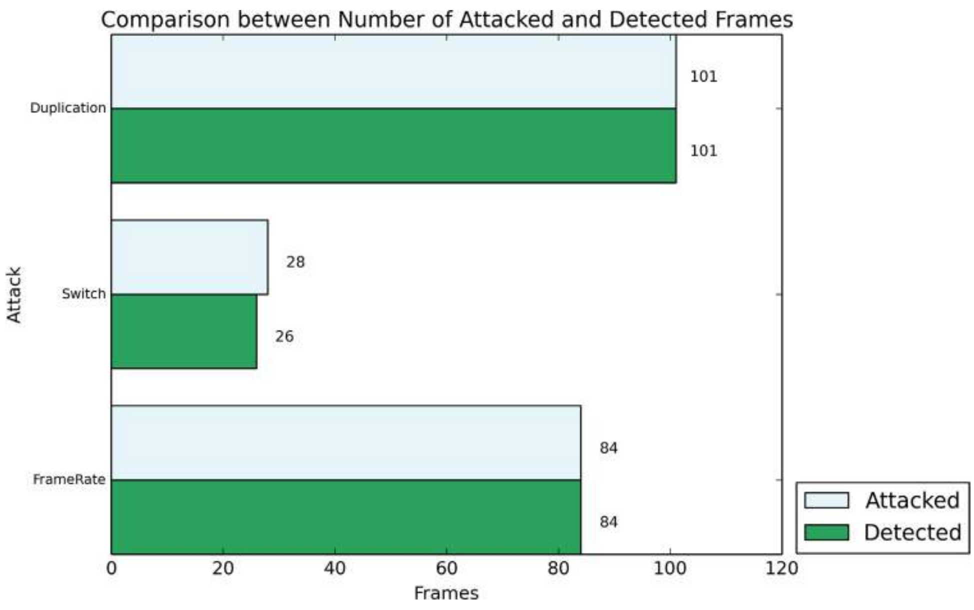

Fig. 13. Number of temporal detected and attacked frames of the ‘Sunset’ video.

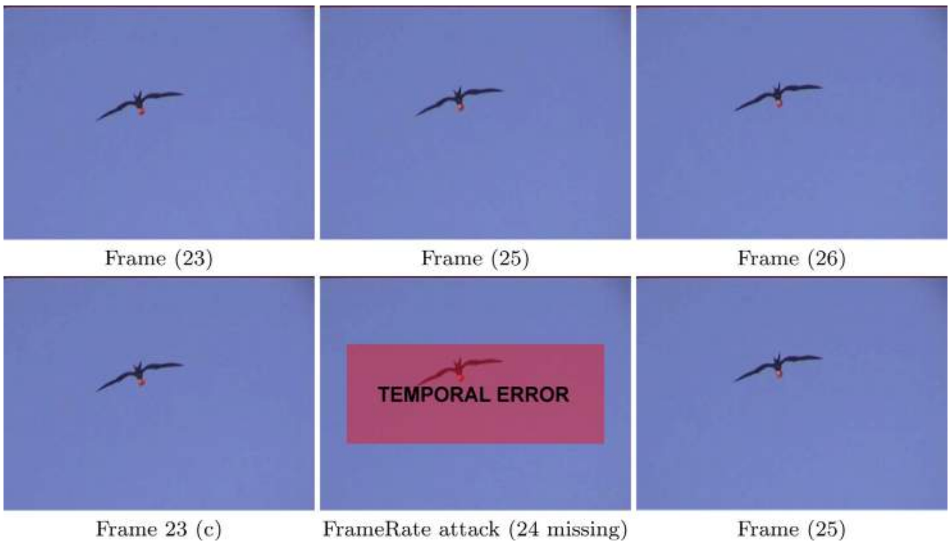

Fig. 14. Temporal FrameRate reduction attack for video ‘Sky’.

Table 7
Mean efficiency of spatial detections per frame for some methods (%).

Method	Spatial attack	Temporal attack
Zhi-yu and Xiang [35]	50.49	–
Hsu et al. [11]	98.34	–
Lin et al. [12]	82.05	–
Pan and Lyu [19]	100	–
Subramanyam and Emmanuel [24]	85.01	99.5
Amerini et al. [1]	98.17	–
Wang and Farid [29]	–	100
Proposed framework	97.86	96.53

Table 7 shows the spatial and temporal efficiencies of the proposed technique in comparison to seven methods available in the literature [1,11,12,19,24,29,35]. In this table, the efficiency is computed as the ratio between the amount of tampered frames and the amount of detected tampered frames. The table shows the percentage of cases in which an attacked frame is detected as being tampered, independent of the size of the attacked areas (detection per frame).

The proposed method is able to detect a larger number of tampering attacks than other methods available in the literature. As shown in Table 7, the proposed method achieves an average detection rate of 97.86% for Spatial Attacks and 96.53% for Temporal Attacks. While [24] and [29] present better results for temporal attacks, [24] has a worse performance for spatial attacks and [29] is not designed to detect them. On the other hand, [1] and [11] have better results for spatial attacks, but are not designed to detect temporal attacks. In summary, most of the methods available in the literature are designed to detect only one specific type of tampering attack. The proposed method uses a single technique to detect several types of tampering attack, obtaining better or equivalent results.

6. Conclusions and future work

In this paper, we presented a framework to detect temporal, global, and local tamperings in digital videos. The proposed framework is based on a simple and fast watermarking algorithm that does not degrade the video quality. It allows identifying the localization of spatial and temporal tampering attacks with pixel granularity. The combination of spatial and temporal marks increases the sensitivity and robustness of the proposed technique. Also, the replication of the temporal mark in the audio and video components makes it possible to identify temporal attacks even when all video content is lost. By analyzing the tampering vector, the framework is also able to determine the type of tampering attack. Overall, when compared with other algorithms, the proposed framework has a good performance, presenting a high accuracy, efficiency, and a low percentage of false-positives and false-negatives.

References

[1] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, G. Serra, A sift-based forensic method for copy–move attack detection and transformation recovery, IEEE Trans. Inf. Forensics Secur. 6 (3) (2011) 1099–1110.
[2] W. Bender, D. Gruhl, N. Morimoto, A. Lu, Techniques for data hiding, IBM Syst. J. 35 (3.4) (1996) 313–336.
[3] O. Cetinkaya, M.L. Koc, Practical aspects of dynavote e-voting protocol, Electron. J. e-Gov. 7 (4) (2009) 327–338.
[4] B. Chen, G.W. Wornell, Quantization index modulation: a class of provably good methods for digital watermarking and information embedding, IEEE Trans. Inf. Theory 47 (4) (2001) 1423–1443.
[5] I.J. Cox, J.-P. M. G. Linnartz, Some general methods for tampering with watermarks, IEEE J. Sel. Areas Commun. 16 (4) (1998) 587–593, doi:10.1109/49.668980.
[6] I.J. Cox, J. Kilian, F.T. Leighton, T. Shamoon, Secure spread spectrum watermarking for multimedia, IEEE Trans. Image Process. 6 (12) (1997) 1673–1687.
[7] D. Cross, B.G. Mobasseri, Watermarking for self-authentication of compressed video, in: Proceedings of International Conference on Image Processing, 2, 2002, pp. II-913–II-916, doi:10.1109/ICIP.2002.1040100.
[8] D.E. Eastlake, S.D. Crocker, J.I. Schiller, Rfc–1750 Randomness Recommendations for Security, Network Working Group, 1994.
[9] P. FIPS, 140-2: Security Requirements for Cryptographic Modules, National Institute of Standards and Technology (2001).
[10] F. Hartung, Digital Watermaking and Fingerprinting of Uncompressed and Compressed Video, Universitat, Erlangen, Nurnberg, 2000 Ph.D. thesis.
[11] C.-C. Hsu, T.-Y. Hung, C.-W. Lin, C.-T. Hsu, Video forgery detection using correlation of noise residue, in: Proceedings of the 10th IEEE Workshop on Multimedia Signal Processing, IEEE, 2008, pp. 170–174.
[12] E. Lin, A.M. Eskicioglu, R.L. Lagendijk, E.J. Delp, Advances in digital video content protection, Proc. IEEE 93 (1) (2005) 171–183.
[13] E.T. Lin, E.J. Delp, Video and Image Watermark Synchronization, Purdue University, West Lafayette, IN, 2005.
[14] B.G. Mobasseri, M.J. Sieffert, R.J. Simard, Content authentication and tamper detection in digital video, in: Proceedings of International Conference on Image Processing, 1, 2000, pp. 458–461vol.1, doi:10.1109/ICIP.2000.900994.
[15] P. Mumby, Reefvid: A Resource of Free Coral Reef Video Clips for Educational Use, 2013. URL http://www.reefvid.org/.
[16] T.-T. Ng, S.-F. Chang, Q. Sun, Blind detection of photomontage using higher order statistics, in: Proceedings of International Symposium on Circuits and Systems, ISCAS, 5, 2004, pp. V-688–V-691.
[17] Oracle, Securerandom Number Generation (RNG) Algorithms, 2014. URL http://docs.oracle.com/javase/7/docs/api/java/security/SecureRandom.html.
[18] A. Ostaszewska, R. Kłoda, Quantifying the amount of spatial and temporal information in video test sequences, in: J. Ryszard, T. Mateusz, S. Roman (Eds.), Recent Advances in Mechatronics, Springer Berlin Heidelberg, 2007, pp. 11–15.
[19] X. Pan, S. Lyu, Region duplication detection using image feature matching, IEEE Trans. Inf. Forensics Secur. 5 (4) (2010) 857–867.
[20] F. Peng, X.-l. Wang, Digital image forgery forensics by using blur estimation and abnormal hue detection, in: Proceedings of Symposium on Photonics and Optoelectronic, SOPO, 2010, pp. 1–4.
[21] M. Pinson, The consumer digital video library [best of the web], IEEE Signal Process. Mag. 30 (4) (2013) 172–174.
[22] J.A. Redi, W. Taktak, J.-L. Dugelay, Digital image forensics: a booklet for beginners, Multimed. Tools Appl. 51 (1) (2011) 133–162.
[23] J.R. Smith, B.O. Comiskey, Modulation and information hiding in images, in: Information Hiding, Springer, 1996, pp. 207–226.

[24] A. Subramanyam, S. Emmanuel, Video forgery detection using hog features and compression properties, in: Proceedings of the 14th IEEE International Workshop on Multimedia Signal Processing, MMSP, IEEE, 2012, pp. 89–94.
[25] R.G. Van Schyndel, A.Z. Tirkel, C.F. Osborne, A digital watermark, in: Proceedings of IEEE International Conference on Image Processing, 2, IEEE, 1994, pp. 86–90.
[26] A.W.A. Wahab, M.A. Bagiwa, M.Y.I. Idris, S. Khan, Z. Razak, M.R.K. Ariffin, Passive video forgery detection techniques: a survey, in: Proceedings of the 10th International Conference on Information Assurance and Security, IAS, IEEE, 2014, pp. 29–34.
[27] Q. Wang, Z. Li, Z. Zhang, Q. Ma, Video inter-frame forgery identification based on consistency of correlation coefficients of gray values, J. Comput. Commun. 2 (04) (2014) 51.
[28] W. Wang, Digital Video Forensics, 2009.
[29] W. Wang, H. Farid, Exposing digital forgeries in video by detecting duplication, in: Proceedings of the 9th workshop on Multimedia & security, ACM, 2007, pp. 35–42.
[30] W. Wang, X. Jiang, S. Wang, M. Wan, T. Sun, Identifying video forgery process using optical flow, Digital-Forensics and Watermarking, Springer, 2014, pp. 244–257.
[31] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 13 (4) (2004) 600–612.
[32] L. Yin, Z. Bai, R. Yang, Video forgery detection based on nonnegative tensor factorization, in: Proceedings of the 4th IEEE International Conference on Information Science and Technology, ICIST, IEEE, 2014, pp. 148–151.
[33] P. Yin, H.H. Yu, Classification of Video Tampering Methods and Countermeasures Using Digital Watermarking, 2001239–246. 10.1117/12.448208URL http://dx.doi.org/10.1117/12.448208.
[34] Z. Zhang, J. Hou, Q. Ma, Z. Li, Efficient video frame insertion and deletion detection based on inconsistency of correlations between local binary pattern coded frames, Secur. Commun. Netw. 8 (2) (2015) 311–320.
[35] H. Zhi-yu, T. Xiang-hong, Integrity authentication scheme of color video based on the fragile watermarking, in: Proceedings of International Conference on Electronics, Communications and Control, ICECC, 2011, pp. 4354–4358, doi:10.1109/ICECC.2011.6067709.