Applied Mathematical Modelling 85 (2020) 273–293

Contents lists available at ScienceDirect

Applied Mathematical Modelling

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

A discrete wavelet transform and singular value decomposition-based digital video watermark method

Qingliang Liu[a], Shuguo Yang[a], Jing Liu[b], Pengcheng Xiong[a], Mengchu Zhou[c,*]

[a] School of Mathematics and Physics, Qingdao University of Science and Technology, Qingdao, 266061, China
[b] Weifang University of Science and Technology, Weifang, 262700, China
[c] Department of ECE, New Jersey Institute of Technology Newark, NJ 07102-1982 USA

ARTICLE INFO

Article history:
Received 23 October 2019
Revised 19 March 2020
Accepted 16 April 2020
Available online 4 May 2020

Keywords:
Video watermark
DWT-SVD
Imperceptibility
Robustness
Distortion

ABSTRACT

Digital video watermark is widely used to protect copyright and content authentication. DWT-SVD based method, i.e., discrete wavelet transform (DWT) and singular value decomposition (SVD) based method, is one of the most popular state-of-the-art methods. There are two main criteria to evaluate it, i.e., imperceptibility and robustness. The former is measured via the peak signal to noise ratio (PSNR) and structural similarity (SSIM). They should be as high as possible after the watermark is embedded. Robustness measures how easy to restore the embedded watermark by the owner even if the watermarked video is damaged by an outside attacker. Current studies randomly choose embedded positions for watermark, which hardly achieves its highest imperceptibility and robustness. In this paper, we propose a more imperceptible and robust digital video watermarking method than the existing one. First, we introduce a method to measure video frame distortion in a wavelet transform domain. Second, we model the problem to achieve the minimum video frame distortion as an optimization problem, i.e., choose embedded positions that can maximize peak signal to noise ratio. Finally, we evaluate and compare our method with the state-of-the-art approaches by using real experiments and show its advantages in both imperceptibility and robustness.

1. Introduction

With the rapid growth of multimedia technology, billions of bits of content are created in a sub-second. Many issues, such as protection of intellectual property rights of the content, proving ownership, and preventing illegal distribution of digital video become more and more significant [1]. There are lots of policy-wise and technology-wise methods. An effective technology-wise method to limit illegal distribution is called digital video watermark where additional watermark information is embedded in digital video [2]. When digital video is copied, watermark remains intact. The video owners can prove the ownership or copyright of data by extracting and testing the watermark from their digital watermarked video. The true owners can always keep their data safe, which makes it very difficult for a counterfeiter to remove or change watermark [3].

* Corresponding author.
E-mail addresses: LQL_2017@163.com (Q. Liu), ysg_2005@163.com (S. Yang), wkyliujing@163.com (J. Liu), xiong.pengcheng@gmail.com (P. Xiong), zhou@njit.edu (M. Zhou).

https://doi.org/10.1016/j.apm.2020.04.015
0307-904X/

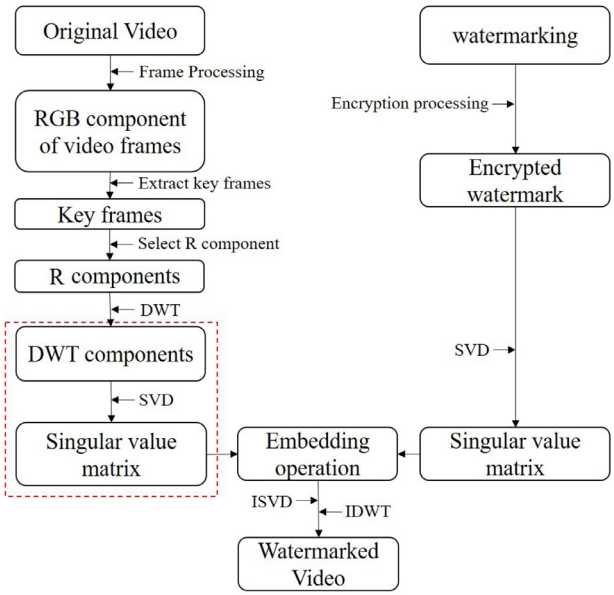


Fig. 1. DWT-SVD-based video watermark algorithm, where a rectangular box represents a result and symbol ' → ' or ' ← ' stands for an action.

According to the domain in which the watermark is embedded, a video watermark algorithm can be classified into spatial-domain methods [4–7] and transform-domain ones [3,8–11]. The former are to embed a watermark into the spatial-domain component of original video frames. They have low complexity and are easy to implement. But they are generally fragile to various attacks, e.g., adding noise, frame averaging and so on [12,13]. The latter are to embed a watermark by modulating the magnitude of coefficients in a video frame transform-domain, such as discrete cosine transform (DCT), discrete wavelet transform (DWT), dual-tree complex wavelet transform (DTCWT), and singular value decomposition (SVD)[14–20]. It is more robust against the attacks mentioned above, but the complexity is higher than spatial-domain watermarking algorithm [14,15]. Specifically, each of these transforms has its own performance for watermarking algorithms. For example, DCT has good energy compaction properties and is used in the most popular compression formats, such as JPEG, MPEG and H.26x [16]. DWT has excellent spatio-frequency localization properties. It is very suitable to identify areas in a digital video frame where watermark can be imperceptibly embedded [17,18]. DCT-based and DWT-based watermarking algorithms are comparatively better than spatial domain watermarking approaches for low-pass filtering, noise addition, sharpening, brightness and contrast adjustments, blurring and compression. However, they are computationally more expensive and provide limited protection against geometric attacks such as scaling, rotation and cropping. DTCWT has better shift invariance than DWT [19,20]. Therefore, it shows better robustness against geometric attacks. SVD is used in a video watermark method because slight variations of singular values do not affect the visual perception of digital video. Usually, it is applied to improve the imperceptibility and robustness of watermark embedding algorithms in combination with other transforms, such as, DWT-SVD-based watermarking [18,21,22], and DTCWT-SVD-based watermarking [23,24].

Recent video watermarking algorithms based on DWT and SVD can be classified into three categories. The first one is to embed watermark in all video frames, i.e., embed same watermark in all frames of video [8,25–29]. They tend to have poor imperceptibility of watermark, poor robustness against video frame dropping, averaging and swapping, and own high time complexity. The second one is to embed watermark in key frames, i.e., embed same watermark in all key frames [13,18,30–32]. Hence the time complexity is greatly reduced. However, the imperceptibility of watermark and robustness under video frame dropping, averaging and swaping is poor. The last one is to embed watermark in key frames according to different video scenes, i.e., embed different watermarks in all key frames according to different video scenes [33]. Its robustness against video attacks is improved, especially under video frame dropping, averaging and swaping. However, its imperceptibility of watermark is poor.

In general, a video watermark method based on DWT and SVD [18,30–33] consists of four steps as shown in Fig. 1, i.e., video preprocessing, watermark preprocessing, watermark embedding and watermarked video reorganization. Video preprocessing is to obtain an R component of key frames after video sequence frame processing and key frame extraction, where the R component is the red component of the RGB (Red, Green, Blue) color space of a key frame. In watermark preprocessing, a watermark is first encrypted, and then SVD is performed in the encrypted watermark to get the singular value matrix. In watermark embedding, DWT is applied on the R component of video key frames, and SVD is performed on DWT components to obtain a singular value matrix. Then, the singular value matrix of DWT components is modified with the singular value matrix of the encrypted watermark. Finally, we obtain the watermarked key frames through inverse SVD (ISVD)

Fig. 2. An example of video frame distortion cost, (a) original video frame, (b) wavelet component coefficients, (c) wavelet coefficient distortion cost, and (d) positions with lowest distortion cost.

and inverse DWT (IDWT). Watermarked video sequence reorganization reconstructs a video sequence with watermarked key frames.

All of the previous video watermark algorithms based on DWT-SVD have poor imperceptibility because they fail to fully consider the effect of different wavelet coefficients on video frame distortion. In other words, they treat the video frame distortion caused by different wavelet coefficients as same. Therefore, if a method is designed to measure the video frame distortion caused by the change of different wavelet coefficients, we can improve the imperceptibility of watermark video while guaranteeing the robustness of watermark algorithm against attacks, i.e., increase peak signal to noise ratio (PSNR) and structural similarity (SSIM) values of watermark video frames and ensure subjectively excellent quality of watermarked video without decreasing normalized cross correlation (NCC).

Specifically, for a video frame as shown in Fig. 2(a), its wavelet coefficients of one DWT component are shown in Fig. 2(b) where these wavelet coefficients (or their corresponding) can be used to embed a watermark. Current algorithms [13,18,30–33] randomly select some wavelet coefficients or select all of them to embed the watermark. In other words, they assume that the video frame distortion caused by each wavelet coefficient change is the same. However, experimental results show that video frame distortion is different when we use different wavelet coefficients in a DWT component to embed a watermark. State-of-the-art algorithms [13,18,30–33] hardly preserve high video quality as measured by high peak signal to noise ratio (PSNR) when they ignore the difference. Even for DTCWT-SVD based methods [23,24], we need to find target component coefficients of DTCWT to embed watermark and minimize the distortion caused by embedding the watermark into the original video.

We transform the problem of how to find target wavelet coefficients into an optimization problem, i.e., to minimize the video frame distortion caused by embedding watermark into these target wavelet coefficients with respect to the size of the watermark. To this end, we propose a video frame distortion cost calculation method based on wavelet coefficients from the DWT component of a video frame. For example, Fig. 2(c) shows the distortion cost corresponding to the wavelet coefficients in Fig. 2(b). The darker positions in Fig. 2(c), the lower the distortion cost, i.e., the video frame distortion caused by changing the wavelet coefficient of these positions is small. Therefore, if we choose these positions correspondent to the wavelet coefficients with lowest distortion cost as shown in Fig. 2(d) to embed the watermark, we can expect to improve the PSNR value of a watermarked video frame compared to those algorithms in [13,18,30–33].

The rest of this paper is organized as follows. In Section 2, we introduce the preliminaries. Next, we give the proposed video watermark algorithm in Section 3 followed by the evaluation in Section 4. Finally, the paper is concluded in Section 5.

2. Preliminaries

In this section, we present the notations used in this article. Then, we introduce the preprocessing of video, which consists of scene division and key frame extraction from original multiple frames, as shown in Fig. 3. Next, we describe watermark preprocessing. Finally, we give an example for video preprocessing and watermark preprocessing, respectively.

2.1. Notations

Throughout the paper, matrices are expressed in capital letter, vectors and sets are written in lowercase letter, unless otherwise specified. In Table 1, we list major notations.

2.2. Video preprocessing

A popular video watermark embedding method is to embed the same watermark into the key frames of a video sequence [13,30,34,35], because this can greatly reduce the time complexity of the algorithm, while causing lower video distortion

Fig. 3. 'suzie.avi' video scene division and key frame extraction.

Table 1
Major notations.

Notation	Description
$\mathbb{N}_k$	The integer set $\mathbb{N}_k = \{0, 1, \ldots, k-1\}$, and k is any positive integer.
$\mathbb{N}_{j\to k}$	The integer set $\mathbb{N}_{j\to k} = \{j, j+1, \ldots, k-1\}$, and $k > j > 1$.
m	Height of a video frame
n	Width of a video frame
L	The number of frames in a video sequence
S	The number of scenes in a video sequence
K	The number of key frames included in a video sequence
K_s	The number of key frames included in the s^{th} video scene, where $s \in \mathbb{N}_S$.
$I_{k,s}$	R component of the k^{th} key frame in the s^{th} video scene, where $k \in \mathbb{N}_{K_s}$
$D_{k,s}$	Distortion cost of the k^{th} key frame in the s^{th} video scene
w_s	Original watermark corresponding to the s^{th} video scene
p	Height of the original watermark
q	Width of the original watermark
$\tilde{w}_s$	Encrypted watermark
$\tilde{I}_{k,s}$	R component of the k^{th} key frame in the s^{th} watermarked video scene

than embedding the same watermark into all video frames. A more secure method [33] is to first divide a video sequence into video scenes and then embed different watermarks into the key frames of different scenes. In this section, we first describe how to divide a video sequence into scenes, and then introduce how to extract key frames from it.

2.2.1. Video scene division

In order to quickly and effectively divide a video sequence into several scenes, we use histogram differences among adjacent frames to divide a scene, i.e.,

$$d(i,j) = \sum_{r=0}^{255} \frac{(h_j(r) - h_i(r))^2}{h_j(r) + h_i(r)}, \tag{1}$$

where $d(i, j)$ represents the histogram difference between the gray frames of the i^{th} and j^{th} video frames, $i \in \{0, 1, \ldots, L-2\}$ and $j = i+1$. Here a gray frame can be obtained from a video frame in an RGB color space, L represents the total number of video frames contained in a video sequence. $h_i(r)$ or $h_j(r)$ is the number of occurrences of pixel grayscale r in the i^{th} or j^{th} video frame, $r \in \mathbb{N}_{256}$.

We normalize the histogram difference as follows:

$$\hat{d}(i,j) = \frac{d(i,j) - min(\mathbb{D})}{max(\mathbb{D}) - min(\mathbb{D})}, \tag{2}$$

where $\mathbb{D} = \{d(i,j) | i = 0, 1, \ldots, L-2;\ j = i+1\}$, $min(\mathbb{D})$ represents the minimum of the elements in set $\mathbb{D}$, $max(d)$ represents the maximum of the elements in set $\mathbb{D}$.

In general, scene change detection is carried out among continuous frames based on histogram differences. We define a video frame i where the video scene changes if:

(i) $\hat{d}(i-1,i) \le \hat{d}(i,i+1)$ and $\hat{d}(i,i+1) \ge \hat{d}(i+1,i+2)$,
(ii) $\hat{d}(i,i+1) \ge \Omega$, where Ω is an artificially preset threshold, and $\Omega \in (0, 1)$, and
(iii) There are at least N video frames between the current video frame i and a previous video frame where the video scene changes, i.e., a video scene must contain at least N video frames. Here N is the number of video frames that a person can perceive in the duration t of human vision [36], i.e.,

$$N = \lceil v \times t \rceil, \tag{3}$$

where $\lceil \cdot \rceil$ is "ceiling function" (e.g., $\lceil 3.1 \rceil = 4$), v is video frame rate, and t is the minimum time length of a video sequence. If the time length of a video sequence is less than t, the human will not notice it. In general, $t \in [0.1, 0.4]$.

Specifically, we set the 0th video frame as an initial video scene change frame. Start with it if the $i^{th}(i > 1)$ video frame meets firstly the above conditions i and ii, and $i > N$, i.e., condition iii is true, then the i^{th} video frame is one where the video scene changes. The 0th video scene should contain video frames with an interval of $\mathbb{N}_i$. Then start with the i^{th} video frame, if the $j^{th}(j > i)$ video frame meets conditions i and ii, and $j - i > N$, then the j^{th} one is the next video frame where the video scene changes. The 1st video scene should have frame video numbered $\mathbb{N}_{i \to j}$ and so on.

2.2.2. Key frame extraction

In order to reduce the time complexity of the algorithm, we simply and quickly extract the key frames through the histogram differences among video frames. Key frame is extracted by the histogram difference between the current key frame and the other video frames. The histogram difference between the i^{th} and j^{th} video frames can be computed via (1) as well. The key frame selection algorithm is shown in Algorithm 1.

Algorithm 1 Key Frame Selection.

Input: All video frames and a threshold value Λ
Output: Key frames
1: Set $i = 0$ and $l = 0$
2: **While** $i + l + 1 < L$ **do**
3: **While** $d(i, i + l + 1) < \Lambda$ **do**
4: $l = l + 1$
5: **end**
6: Store i^{th} video frame in video key frames list.
7: $i = i + l + 1$
8: $l = 0$
9: **end**
10: Return the key frames

In Algorithm 1 Λ is a preset threshold histogram difference and n is the total number of key frames. We can distribute these extracted key frames into the video scenes.

2.3. Watermark preprocessing

In order to improve its security, a watermark should be encrypted before embedding. This work selects the Arnold transformation encryption algorithm [37] which is a commonly used scrambling encryption method. It can reduce the correlation between adjacent pixels of a watermark. Arnold transform is essentially the scrambling of video frame pixel coordinates. In addition, the watermatk matrix must be square to use it.

For a watermark with the size of $p \times q$ with $p = q$, and given coordinate $Y = [x\ y]'$, $(x, y \in \mathbb{N}_p)$, we can get the coordinate $Y^* = [x^*\ y^*]'$ after r times Arnold transformation, i.e.,

$$Y^* = A^r Y \ mod \ p, \tag{4}$$

where Y and Y^* are the initial and the transformed coordinates, respectively.

$A = \begin{pmatrix} 1 & a \\ b & ab+1 \end{pmatrix}$, a and b are any positive integers. In this paper, let's set $a = b = 1$, $r = 10$, and then $A^{-1} = \begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix}$.
The inverse Arnold transformation is:

$$Y = (A^{-1})^r Y^* \ mod \ p, \tag{5}$$

where r can be any positive integer.

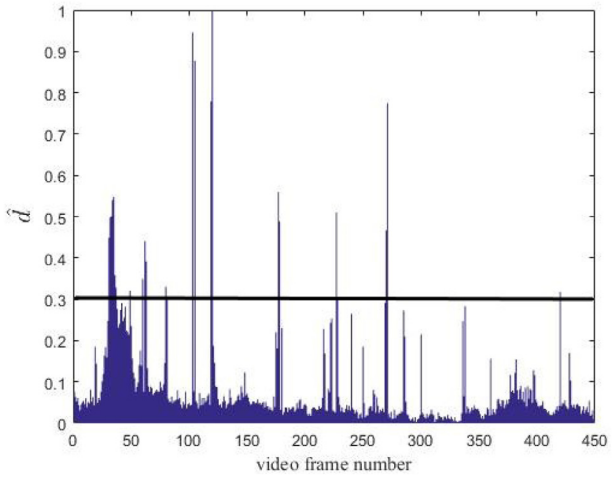


Fig. 4. Scene change detection using a histogram difference method.

Table 2
Key frames from suzie.avi.

Scene(contains the frames)	Key frames
*Scene*0 ($l \in \mathbb{N}_{34}$)	0,17,25,29,31,32,33
*Scene*1 ($l \in \mathbb{N}_{34\to 48}$)	34,35,37,39,42,44,46
*Scene*2 ($l \in \mathbb{N}_{48\to 79}$)	48,50,56,60,67,76
*Scene*3 ($l \in \mathbb{N}_{79\to 102}$)	80,86
*Scene*4 ($l \in \mathbb{N}_{102\to 119}$)	102,103,105
*Scene*5 ($l \in \mathbb{N}_{119\to 176}$)	119,120,145,154,175
*Scene*6 ($l \in \mathbb{N}_{176\to 226}$)	225
*Scene*7 ($l \in \mathbb{N}_{226\to 270}$)	227,232,269
*Scene*8 ($l \in \mathbb{N}_{270\to 419}$)	300,373,382,394
*Scene*9 ($l \in \mathbb{N}_{420\to 450}$)	420,443

In particular, watermark mentioned in this paper is binary, i.e., it has only black and white pixels.

2.4. Examples

2.4.1. Video preprocessing

We take suzie.avi [33] as a video example, which can be downloaded from http://trace.eas.asu.edu/. Its frame rate is 29.97 frames per second and it contains 450 video frames, as partially shown in Fig. 3.

First, we get normalize the histogram difference $\hat{d}$, as shown in Fig. 4.

Then we further divide this video into 10 scenes based on $\hat{d}$ according to the algorithm in Section II.B. We set the threshold $\Omega = 0.3$ and the duration of human vision $t = 0.4$, i.e., the number of video frames that a person can perceive in the duration of human vision is $N = \lceil 29.97 \times 0.4 \rceil = 12$. Specifically, we find $\hat{d}(33, 34) \le \hat{d}(34, 35)$, $\hat{d}(35, 36) \le \hat{d}(34, 35)$, and $\hat{d}(34, 35) \ge 0.3$, and this part contains 35 video frames from the 0th to 34th video frame, which is greater than $N = 12$. So the 34th video frame is a video scene change frame, i.e., *Scene*0 should contain the count of video frames with a set of $\mathbb{N}_{34}$. Similarly, we can determine the interval of video frames contained in *Scene*1 – *Scene*9, respectively, as shown in Fig. 3 and Table 2. Finally, we extract the key frames of suzie.avi according to Algorithm 1, in which $\Lambda = 2400$. We divide these extracted key frames according to the above video scenes, as shown in Fig 3 and Table 2.

2.4.2. Watermark preprocessing

Given a watermark w_s, we can obtain the encrypted watermark $\tilde{w}_s$ according to (3). We take 'QUST' binary video frame of size 64×64 as an example, as shown in Fig. 5(a), and its orignal video frame can be downloaded from https://www.qust.edu.cn/xywh/UIS/sjsbxt_VI_.htm. When $r = 10$, the encryped watermark after Anorld transformation is shown in Fig. 5(b).

3. Video watermark algorithm

Due to its excellent spatio-frequecy location properties, DWT is widely used in video watermarking [18]. In this section, we first introduce several important metrics and methods to evaluate the video watermarking scheme, i.e., PSNR (peak signal to noise ratio), SSIM (structural similarity), DSCQS (double stimulus continuous quality scale) and NCC (normalized

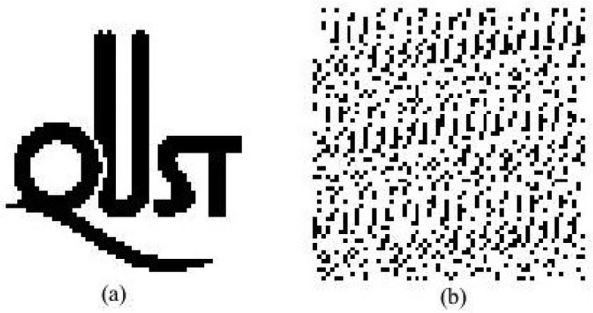


Fig. 5. Watermark encryption with Arnold transformation, (a) original watermark, and (b) encrypted watermark.

cross correlation). We then show how to calculate the wavelet coefficient distortion cost of a video frame. Finally, we present an optimization model and its solution algorithms.

3.1. Evaluation metrics and methods

The evaluation methods of watermark video quality are divided into objective and subjective evaluations. For the former, we generally choose PSNR [33] and SSIM [38], i.e.,

$$PSNR = \frac{1}{L}\sum_{l=0}^{L-1}\left(20 \times lg\frac{255}{\sqrt{MSE_l}}\right), \tag{6}$$

where L represents the number of video frames, MSE_l is the mean square error, i.e.,

$$MSE_l = \frac{1}{mn}\sum_{k=0}^{2}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\|I_l(i,j,k) - I_l^*(i,j,k)\|, \tag{7}$$

where m and n are the height and width of a video frame, respectively. $I_l(i, j, k)$ and $I_l^*(i, j, k)$ are the pixel matrices of the original and watermarked frame, respectively.

$$SSIM = \frac{1}{L}\sum_{l=0}^{L-1}F(I_l(i,j,k), I_l^*(i,j,k)), \tag{8}$$

where function F is used to calculate the structural similarity between the original frame I_l and the watermarked one I_l^*, i.e.,

$$F(I_l, I_l^*) = \frac{(2\mu_{I_l}\mu_{I_l^*} + c_1)(2\sigma_{I_l,I_l^*} + c_2)}{(\mu_{I_l}^2 + \mu_{I_l^*}^2 + c_1)(\sigma_{I_l}^2 + \sigma_{I_l^*}^2 + c_2)}, \tag{9}$$

where μ_{I_l} and $\mu_{I_l^*}$ are the mean of I_l and I_l^*, $\sigma_{I_l}^2$ and $\sigma_{I_l^*}^2$ are the variance of I_l and I_l^*, σ_{I_l,I_l^*} is the covariance of I_l and I_l^*, and c_1 and c_2 are constants ($c_1 = (255k_1)^2, c_2 = (255k_2)^2, k_1 = 0.01, k_2 = 0.03$). In general, the larger PSNR or SSIM, the better watermarked frame quality.

For subjective evaluation, we often choose the DSCQS [39] method. In DSCQS, a group of observers who receive special training are required to score an original and a watermarked version of a test frame where the observers do not know the labels. During the scoring, the original and watermarked frames are displayed alternately more than twice, and the order of different test frames that are presented to the observers is pseudo-random. Finally, a reasonable quality result is given according to the score of the observers. The quality result is divided into five grades: excellent, good, medium, poor and inferior. More details about DSCQS can be found in [39].

NCC (normalized cross correslation) is used to verify the validity of watermark extracted from a watermarked video sequence [35], i.e.,

$$NCC = \frac{\sum_{i=0}^{p-1}\sum_{j=0}^{q-1}(w(i,j) \times w^*(i,j))}{\sqrt{\sum_{i=0}^{p-1}\sum_{j=0}^{q-1}(w(i,j))^2}}, \tag{10}$$

where $w(i, j)$ and $w^*(i, j)$ are the original and extracted watermark pixel matrix, respectively. The integrity of extracted watermark and robustness can be calculated via NCC. When NCC approaches 1, not only the extracted watermark information becomes complete and easier to distinguish, but also the robustness of a watermarking algorithm is better.

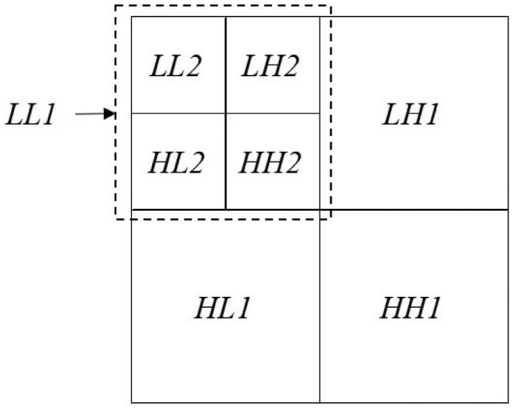


Fig. 6. Video frames DWT, *LL1, LH1, HL1* and *HH1* represent the DWT components at level 1, and *LL2, LH2, HL2* and *HH2* are DWT components at level 2.

3.2. Wavelet coefficient distortion cost for key frames

According to (6)–(9), we know that the PSNR and SSIM values of a watermarked video frame is related to the difference between the original and watermarked video frames. This difference is due to the distortion of the original one caused by the change of wavelet coefficients when the watermark is embedded into it. However, video frame distortion caused by embedding watermark into different wavelet coefficients is different. So next we develop a method to compute wavelet coefficient distortion cost for key frames.

3.2.1. Discrete wavelet transform video key frame

Given an R component from the k^{th} key frame in the s^{th} video scene, noted as $I_{k,s}, k \in \mathbb{N}_{K_s}, s \in \mathbb{N}_S$. In general, a video frame 2-D DWT requires a 2-D scaling function $\varphi(x, y)$ and three 2-D wavelet functions $\psi(x, y)$ [40]. Wavelet transform functions are as follows:

$$\begin{cases} \varphi(x,y) = \varphi(x) \otimes \varphi(y) \\ \psi^0(x,y) = \varphi(x) \otimes \psi(y) \\ \psi^1(x,y) = \psi(x) \otimes \varphi(y) \\ \psi^2(x,y) = \psi(x) \otimes \psi(y) \end{cases} \tag{11}$$

where ψ^0, ψ^1 and ψ^2 represent the functions for horizontal, vertical and diagonal directions, respectively. Operator $\otimes$ represents convolution.

A video frame can be divided into a lower resolution approximation video frame (*LL*) and three detailed components, vertical (*LH*), diagonal (*HH*) and horizontal (*HL*), Video frame DWT method is shown as follows:

$$\begin{cases} C_{LL}^{k,s} = I_{k,s} \otimes \varphi \\ C_{LH}^{k,s} = I_{k,s} \otimes \psi^0 \\ C_{HL}^{k,s} = I_{k,s} \otimes \psi^1 \\ C_{HH}^{k,s} = I_{k,s} \otimes \psi^2 \end{cases} \tag{12}$$

where the coefficient matrices of *LL, LH, HL* and *HH* components are $C_{LL}^{k,s}$, $C_{LH}^{k,s}$, $C_{HL}^{k,s}$ and $C_{HH}^{k,s}$, respectively. If the dimension of $I_{k,s}$ is $m \times n$, then the dimensions of $C_{LL}^{k,s}, C_{LH}^{k,s}, C_{HL}^{k,s}$ and $C_{HH}^{k,s}$ are $(m/2) \times (n/2)$.

(12) shows a single level wavelet transform. We can continue DWT on the low frequency component *LL* to get the next level wavelet transform if needed. The higher the levels, the more stability of the wavelet. But the number of positions that we can embed watermark also decreases because the area decreases. For example, Fig. 6 shows the diagram of video frame 2-level DWT, where *LL1, LH1, HL1* and *HH1* represent the 1st level's DWT components, *LL2, LH2, HL2* and *HH2* are the 2nd level's DWT components which can be obtained by applying DWT to *LL1* components.

In addition, the choice of a different wavelet base has a great influence on the performance of a watermark algorithm [41]. This work selects Haar wavelet to do DWT for video frames, because it can make a watermark algorithm more robust to attacks than others [42].

3.2.2. Distortion cost

In order to measure the effect of wavelet coefficient changes, we develop a distortion cost method based on coefficients of wavelet components from a video frame.

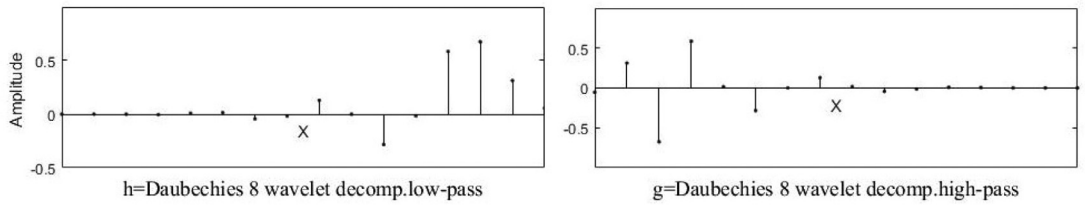


Fig. 7. Selected 1-D low-pass filter h and high-pass filter g.

First, we apply a high-pass decomposition filter g and a low-pass decomposition filter h to construct wavelet directional filter [43], as shown in Fig. 7. Three direction filters at 0°, 45° and 90° where F_0, F_1, and F_2 are:

$$\begin{cases} F_0 = h \cdot g^T \\ F_1 = g \cdot h^T \\ F_2 = g \cdot g^T \end{cases} \tag{13}$$

Second, we calculate the filter residual $R_{k,s} = \{R^0_{k,s}, R^1_{k,s}, R^2_{k,s}\}$ of the wavelet coefficient matrix $C_{k,s}$:

$$\begin{cases} R^0_{k,s} = C_{k,s} \otimes F_1 \\ R^1_{k,s} = C_{k,s} \otimes F_2 \\ R^2_{k,s} = C_{k,s} \otimes F_3 \end{cases} \tag{14}$$

where $C_{k,s} \in \{C^{k,s}_{LL}, C^{k,s}_{LH}, C^{k,s}_{HL}, C^{k,s}_{HH}\}$, and symbol $\otimes$ represents convolution.

Next, we calculate adaptability $A_{k,s}$ of filter residual $R_{k,s}$, where $A_{k,s} = \{A^0_{k,s}, A^1_{k,s}, A^2_{k,s}\}$ and

$$\begin{cases} A^0_{k,s} = \frac{1}{R^0_{k,s}+1} \otimes |F_0|^{180^\circ} \\ A^1_{k,s} = \frac{1}{R^1_{k,s}+1} \otimes |F_1|^{180^\circ} \\ A^2_{k,s} = \frac{1}{R^2_{k,s}+1} \otimes |F_2|^{180^\circ} \end{cases} \tag{15}$$

where $|F_0|^{180^\circ}$, $|F_1|^{180^\circ}$ and $|F_2|^{180^\circ}$ indicate that the absolute value matrix of the directional filter is rotated 180° counter-clockwise. For example, $|[1, -2, 3, -4]|^{180^\circ} = [4, 3, 2, 1]$.

Finally, we obtain the distortion cost $D_{k,s}$ as the sum of all the adaptability in all directions, i.e.,

$$D_{k,s} = A^0_{k,s} + A^1_{k,s} + A^2_{k,s}. \tag{16}$$

where $D_{k,s}$ corresponds to $C_{k,s}$, and the dimension of $D_{k,s}$ is $(m/2) \times (n/2)$.

The 4th suzie.avi scene contains three key frames, 102nd, 103rd, 105th frames, as shown in Figs. 8(a)–(c), respectively. Their R components are shown in Figs. 8(d)–(f), respectively. Their *LH* wavelet components are shown in Figs. 8(g)-(i), respectively. Video frames distortion cost corresponding to *LH* wavelet components are shown in Figs. 8(j)-(l), respectively. Note that the sizes of C and D are $(m/2) \times (n/2)$ where the original frame size is $m \times n$. If we do 2-level DWT, the size becomes even smaller, i.e., $(m/4) \times (n/4)$.

3.3. Target wavelet coefficient selection

Given an original key frame, we choose its R component, i.e., $I_{k,s}$, which represents the R component of the k^{th} key frame in the s^{th} video scene. First, we perform single level DWT on $I_{k,s}$ to get its *LH* wavelet component $C_{k,s}$. Then, this video frame distortion cost for the *LH* wavelet component is obtained according to (13)–(16).

Next, in order to obtain the best quality of video after embedding a watermark in the *LH* wavelet component, we need to find target *LH* wavelet component coefficients to embed a watermark. To this end, we transform the problem of how to find the target wavelet coefficients into an optimization problem. We assume that the watermark matrix embedded in the s^{th} video scene is w_s, and the size is $p \times q$. Our goal is to find target wavelet coefficients $\hat{C}_{k,s}(b), b \in \mathbb{N}_{p\times q}$, which realize the minimum distortion of this video frame when it changes. Therefore our goal is to find the target distortion cost $\hat{D}_{k,s}(b)$ corresponding to the target wavelet coefficients $\hat{C}_{k,s}(b)$, which have the minimum sum of distortion costs. The optimization problem is formulated as:

$$\textit{Minimize} \qquad f(\hat{D}_{k,s}) = \sum_{b=0}^{p\times q-1} \hat{D}_{k,s}(b), \tag{17}$$

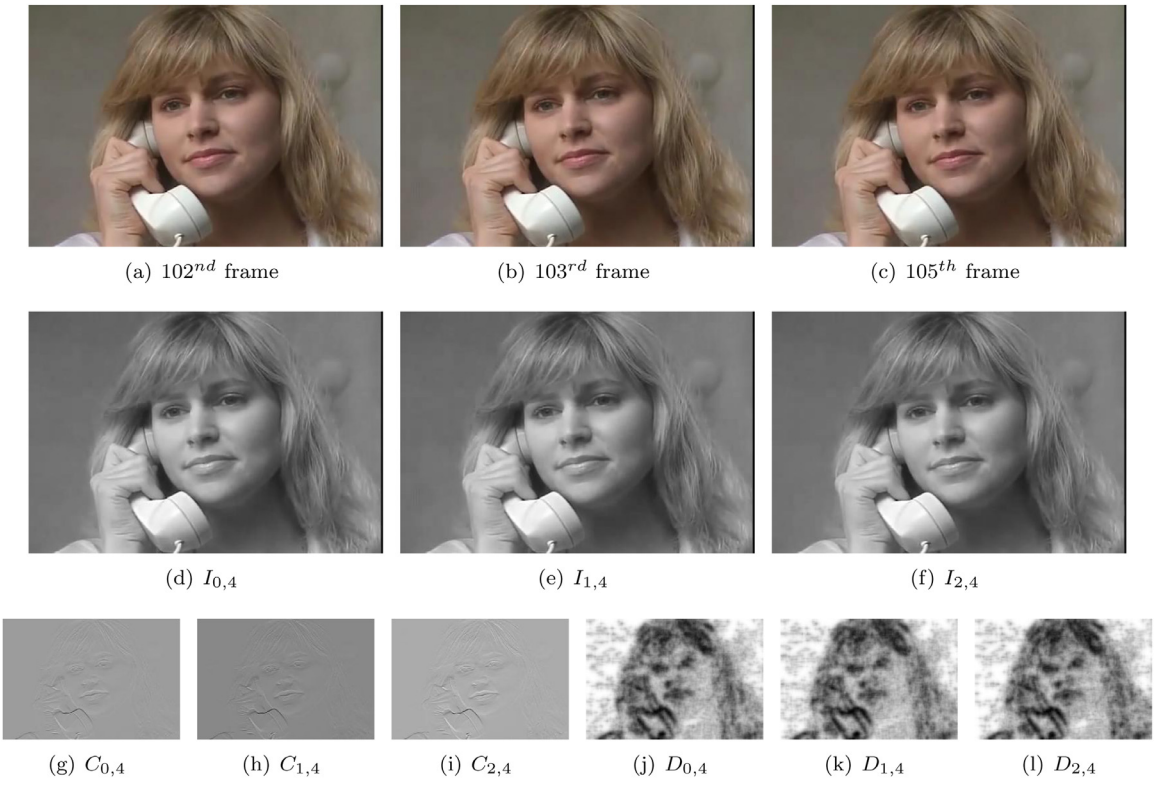


Fig. 8. Video key frames from the 4th suzie.avi scene, R components $I_{0,4}$, $I_{1,4}$ and $I_{2,4}$, *LH* wavelet component coefficients $C_{0,4}$, $C_{1,4}$ and $C_{2,4}$ and wavelet coefficient distortion cost $D_{0,4}$, $D_{1,4}$ and $D_{2,4}$.

where $\hat{D}_{k,s}(b) \in \{D_{k,s}(x,y) \mid x \in \mathbb{N}_{m/2}, y \in \mathbb{N}_{n/2}\}$. We save the coordinates $Z_{k,s}$ of $\hat{D}_{k,s}(b)$ in $D_{k,s}$, i.e., $Z_{k,s} \in \{(x_0,y_0),(x_0,y_1),\ldots,(x_0,y_{p\times q-1}),(x_1,y_0),\ldots,(x_{p\times q-1},y_{p\times q-1})\}$. In fact, (15) is to find $\hat{D}_{k,s}(b)$, which is the set of the top $p \times q$ elements in $D_{k,s}$ with the smallest value.

According to Section III.B, we know each wavelet coefficient correspondings to a distortion cost value. In other word, a wavelet coefficient $C_{k,s}(x, y)$, only corresponds to a distortion value $D_{k,s}(x, y)$. So we can get the target *LH* wavelet component coefficient $\hat{C}_{k,s}(b)$ according to the coordinate $Z_{k,s}$ obtained from the target distortion cost $\hat{D}_{k,s}(b)$. In addition, we generate a matrix $Q_{k,s}$ with the size of $p \times q$ according to $\hat{C}_{k,s}(b)$ for the convenience of the next work, where $Q_{k,s} = R(\hat{C}_{k,s}(b), p, q)$, R is a function which returns a matrix with the size $p \times q$ whose elements are taken columnwise from $\hat{C}_{k,s}(b)$.

For example, if $C_{k,s}(x,y) = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{pmatrix}$, $D_{k,s}(x,y) = \begin{pmatrix} 0.1 & 0.3 & 0.6 \\ 0.5 & 0.9 & 0.7 \\ 0.8 & 0.4 & 0.2 \end{pmatrix}$, where $s \in \mathbb{N}_S, k \in \mathbb{N}_{K_s}$, and $p = q = 2$. We can see $\hat{D}_k(b) = (0.1\ 0.2\ 0.3\ 0.4)$ as those are the smallest values in$D_{k,s}$.

We have coordinates $Z_{k,s} = \{(0,0),(2,2),(0,1),(2,1)\}$ and $\hat{C}_k(b) = [0\ 8\ 1\ 7]$. Then, we reshape $\hat{C}_{k,s}(b)$ to have $Q_{k,s}$ where $Q_{k,s} = R(\hat{C}_{k,s}(b), p, q) = \begin{pmatrix} 0 & 1 \\ 8 & 7 \end{pmatrix}$.

Following the example from the last section, we can get the coordinate Z obtained from the target distortion cost $\hat{D}_{k,s}(b)$ when the size of the watermark is 64 × 64. It is the area marked in Figs. 9(g)–(i). Figs. 9(j)–(l) show the $Q_{k,s}$ video frame generated by target wavelet coefficient $\hat{C}_{k,s}$.

We apply Algorithm 2 to $I_{k,s}$ to obtain a watermarked R component from all key frames. Then, we can get watermarked key frames by replacing original R component $I_{k,s}$ with the watermarked one $\bar{I}_{k,s}$ in key frames. Further, we replace the video key frame in the original video with the watermarked key frame to obtain watermarked video.

3.4. Watermark embedding and extraction algorithm

In this section, we first introduce a watermark embedding algorithm, and then give a watermark extraction algorithm.

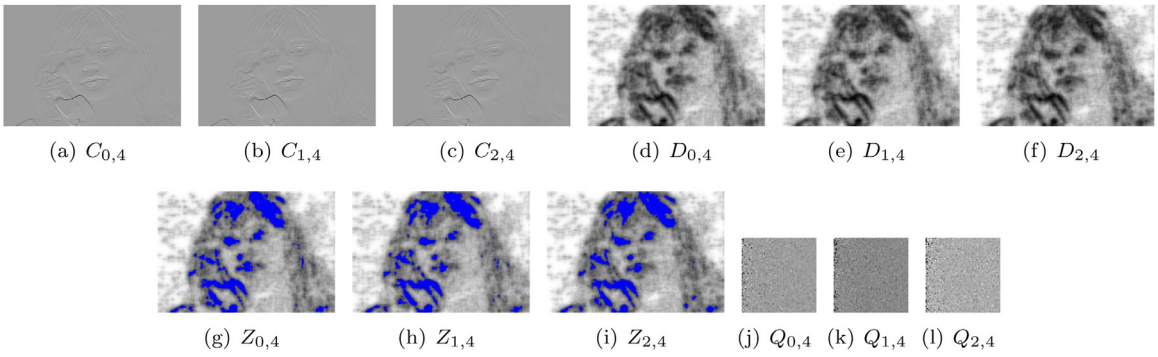


Fig. 9. *LH* wavelet component coefficients $C_{0,4}$, $C_{1,4}$ and $C_{2,4}$, wavelet coefficient distortion costs $D_{0,4}$, $D_{1,4}$ and $D_{2,4}$, target coordinates $Z_{0,4}$, $Z_{1,4}$ and $Z_{2,4}$, and target wavelet coefficients $Q_{0,4}$, $Q_{1,4}$ and $Q_{2,4}$ of key frames.

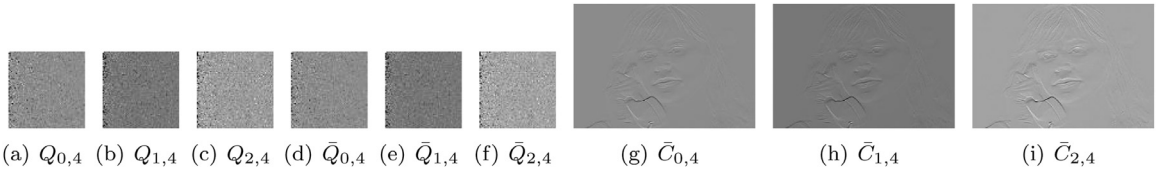


Fig. 10. Target wavelet coefficients $Q_{0,4}$, $Q_{1,4}$, and $Q_{2,4}$, watermarked target wavelet coefficients $\bar{Q}_{0,4}$, $\bar{Q}_{1,4}$, and $\bar{Q}_{2,4}$ and watermarked *LH* wavelet component coefficients $\bar{C}_{0,4}$, $\bar{C}_{1,4}$, and $\bar{C}_{2,4}$.

Algorithm 2 Embed the watermark w_s into $I_{k,s}$.

Input: The R component $I_{k,s}$ of k^{th} key frames from the s^{th} video scene, encrypted watermark video frame $\hat{w}_s$

Output: The watermarked R component $\bar{I}_{k,s}$ of k^{th} key frames from the s^{th} video scene

1: Obtain matrix $Q_{k,s}$ based on $I_{k,s}$ according to section III.B-III.C

2: Perform SVD in the matrix $\hat{w}_s$ and $Q_{k,s}$ as

$$\hat{w}_s = U_s \times S_s \times V_s^T,$$
$$Q_{k,s} = Uo_{k,s} \times So_{k,s} \times Vo_{k,s}^T,$$

where U_s, V_s^T, $Uo_{k,s}$ and $Vo_{k,s}^T$ are all unitary matrix, the dimensions of these matrices are $p \times p$, $q \times q$, $\frac{m}{2} \times \frac{m}{2}$ and $\frac{n}{2} \times \frac{n}{2}$, respectively. The matrices S_s and $So_{k,s}$ are all the singular value matrix, and the size of these are $p \times q$ and $\frac{m}{2} \times \frac{n}{2}$.

3: Modify the singular value of the matrix $Q_{k,s}$ with the singular value of the w_s as

$$\bar{So}_{k,s} = So_{k,s} + \alpha S_s.$$

4: Obtain the modified matrix $\bar{Q}_k$ as

$$\bar{Q}_{k,s} = Uo_{k,s} \times \bar{So}_{k,s} \times V_k^T.$$

5: Generate watermaked *LH* wavelet component $\bar{C}_{k,s}$ based on $Z_{k,s}$ by using $\bar{Q}_{k,s}$. Where $Z_{k,s}$ can be obtained by the section III.C.

6: Apply IDWT on the $\bar{C}_{k,s}$ to obtained the watermarked R component $\bar{I}_{k,s}$.

7: Return $\bar{I}_{k,s}$.

3.4.1. Watermark embedding

In order to improve the robustness of a watermark algorithm, we perform SVD on the matrix of the encrypted watermark and target coefficients matrix of a wavelet *LH* component from video key frames beacause of the stability of SVD against video attacks. In addition, we embed different watermarks into key frames of different video scenes because this can improve the ability of the algorithm against video frame dropping, averaging and swapping. Our specific embedding method is realized in Algorithm 2. In it, α is a scaling factor, to be discussed in Section 4.1.

Following the example shown in Fig. 8, we embed the encrypted watermark as shown in Fig. 5(b) into $Q_{k,s}$ to get $\bar{Q}_{k,s}$ when $\alpha = 1.4$, as shown in Figs. 10(a)–(f). Figs. 10(g)–(i) show the watermaked *LH* wavelet component corresponding to Figs. 9(d)–(f), respectively. Then we can get the watermarked R component $\bar{I}_k$, as shown in Figs. 11(d)–(f), and their corresponding watermarked video key frames in Figs. 11(g)–(i), respectively.

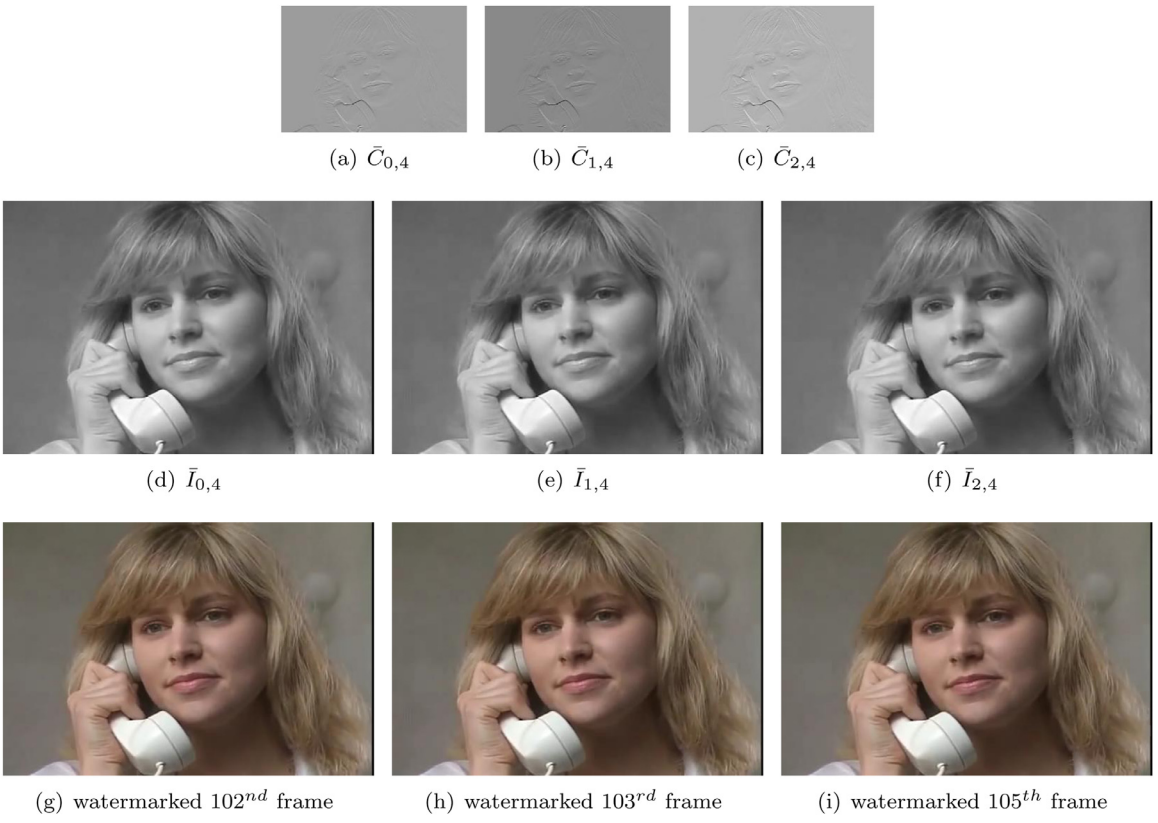


Fig. 11. Watermarked *LH* wavelet component coefficients $\bar{C}_{0,4}, \bar{C}_{1,4}$, and $\bar{C}_{2,4}$, watermarked R components $\bar{I}_{0,4}, \bar{I}_{1,4}$, and $\bar{I}_{2,4}$ and watermarked frames from the 4th suzie.avi scene.

3.4.2. Watermark extraction

A watermark extraction algorithm performs the inverse processing of an embedding algorithm. First, we define a private key exclusive to the original video owner, to be used to extract the watermark, where private key includes the key frame count of the original video scene, target coordinates $Z_{k,s}$, scaling α, singular matrix $So_{k,s}$, matrix $Us_{k,s}$ and matrix $Vs_{k,s}$.

Then, given a video V to be tested, we extract the key frame of V according to the video key frame count of the original video. We can get the R component from these key frames of V. We use $\tilde{I}_{k,s}$ to represent the R component of the k^{th} key frame in the s^{th} scene from V. Next we can extract the watermark $\bar{w}_{k,s}$ from $\tilde{I}_{k,s}$, and this extraction method is implemented in Algorithm 3.

Algorithm 3 Extract the watermark $\bar{w}_{k,s}$ from the $\tilde{I}_{k,s}$.

Input: The R component $\tilde{I}_{k,s}$ of k^{th} watermarked video key frames from the s^{th} watermarked video scene and private key.

Output: Extracted watermark $\bar{w}_{k,s}$ from the s^{th} watermarked video scene

1: Obtain matrix $\tilde{Q}_{k,s}$ based on $\tilde{I}_{k,s}$ according to Section III.B-III.C

2: Perform SVD in the matrix $\tilde{Q}_{k,s}$ as

$$\tilde{Q}_{k,s} = \tilde{U}_{k,s} \times \tilde{S}_{k,s} \times \tilde{V}^T_{k,s}.$$

3: The singular value $\tilde{S}w_{k,s}$ of the watermark can be extracted as

$$\hat{S}w_{k,s} = (\tilde{S}_{k,s} - So_{k,s})/\alpha.$$

4: Encrypted watermark in the k^{th} key frame from the s^{th} video scene to be tested can be obtained as

$$\hat{m}_{k,s} = U_s \times \hat{S}_{k,s} \times V^T_s.$$

5: Decrypt $\hat{m}_{k,s}$ by the formula (**??**) to get $\bar{w}_{k,s}$.

6: Return $\bar{w}_{k,s}$.

Table 3
The value of NCC and PSNR for different scaling factor α.

α	1.1	1.2	1.3	1.4	1.5
PSNR	67.2469	66.6688	66.1334	**65.6004**	65.1380
NCC	0.9990	0.9997	0.9997	**1.0000**	1.0000

Generally speaking, the binary watermark $\bar{w}_{k,s}$ can be extracted from a non-attacked $\tilde{I}_{k,s}$ via Algorithm 3, and it is the same as the watermark extracted from others in the same scene. However, the $\tilde{I}_{k,s}$ usually is attacked. Hence we need to do the following two steps for the $\bar{w}_{k,s}$ obtained by Algorithm 3.

The first step is watermark binarization which ensures that the watermark $\bar{w}_{k,s}$ extracted from an $\tilde{I}_{k,s}$ with attacks is binary, i.e.,

$$\begin{cases} \bar{w}_{k,s} = 0 & if \quad \bar{w}_{k,s} > \Gamma \\ \bar{w}_{k,s} = 1 & else \end{cases} \tag{18}$$

where Γ is a global threshold obtained by using Otsu's method [44]. Specifically, we can get it by using the function 'graythresh' in MATLAB. This function uses Otsu's method, which chooses the threshold to minimize the intraclass variance of the thresholded black and white pixels.

Another step is watermarks integration. We can get multiple similar watermarks via Algorithm 3 and (18) in the same scene as we embed the same watermark in the key frames in the same scene. We integrate all these watermarks to get the final watermark $\bar{w}_s$ that we embed in the scene, i.e.,

$$\tilde{w}_s = G(\bar{w}_{0,s}, \bar{w}_{1,s}, \bar{w}_{2,s}, \ldots, \bar{w}_{K_s-1,s}) \tag{19}$$

where $\bar{w}_{0,s}, \bar{w}_{1,s},$ and $\ldots, \bar{w}_{K_s-1,s}$ are the extracted watermarks from watermarked key frames in the s^{th} scene of a video sequence to be tested, K_s is the number of key frames included in the s^{th} video scene, and the function $G(*)$ represents the mode of pixel value at each pixel position of matrix $\bar{w}_0, \bar{w}_1, \ldots,$ and $\bar{w}_{K_s-1}$. For example, let $\bar{w}_0 = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$, $\bar{w}_1 = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$, $\bar{w}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Then $\tilde{w}_s = G(\bar{w}_0, \bar{w}_1, \bar{w}_2) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$.

For example, first, we show the watermarked key frames from the 4th scene of the video to be tested in Figs. 12(a)–(c), their corresponding watermarked R components $\tilde{I}_{0,4}$, $\tilde{I}_{1,4}$ and $\tilde{I}_{2,4}$ are shown in Figs. 12(d)–(f), respectively. Then, Figs. 12(g)–(i) show watermarked *LH* wavelet components $\tilde{C}_{0,4}$, $\tilde{C}_{2,4}$ and $\tilde{C}_{2,4}$ from the 4th scene of V, and the video frame of $\tilde{Q}_{0,4}$, $\tilde{Q}_{0,4}$ and $\tilde{Q}_{0,4}$ based on $\tilde{C}_{0,4}$, $\tilde{C}_{0,4}$ and $\tilde{C}_{0,4}$ are shown in Figs. 12(j)–(l). Finally, Fig. 12(p) is the extracted watermark $\tilde{w}_4$ obtained by combining $\bar{w}_{0,4}$, $\bar{w}_{1,4}$ and $\bar{w}_{2,4}$ with (19), where $\bar{w}_{0,4}$, $\bar{w}_{1,4}$ and $\bar{w}_{2,4}$ are watermarks from the watermarked key frames, as shown in Figs. 12(m)–(o).

4. Evaluation

4.1. Experiment setups

4.1.1. Software and hardware

Our evaluation is done using MATLAB 9.0 on a server with a 2.8GHZ Intel Core i5-8400 CPU and 8GB memory. All the reported values are obtained by averaging tthe results of 100 simulation experiments.

4.1.2. Selection of α

In Algorithms 2 and 3, α is a scaling factor used to control both imperceptibility and robustness of an embedding algorithm. Generally speaking, bigger α implies worse imperceptibility of watermark but better robustness of watermarked video sequence under no attack. However, there is an upper limit of 1.0 for NCC when α increases under no attack. We usually choose the value when NCC first reaches its upper limit for α.

Specifically, following the example shown in Fig. 8, we embed the encrypted watermark in Fig. 5(b) into key frames. In Table 3, we present PSNR and NCC values for this example with different scaling factors α. When α increases, PSNR of the watermarked video sequence decreases and NCC of the watermark increases until it converges to 1.0. In this example, as NCC first reaches 1.0 at $\alpha = 1.4$, we choose $\alpha = 1.4$.

4.1.3. State-of-the-art methods

We compare our proposed algorithm with two sate-of-the-art methods [13,33]. Faragallah proposes an efficient video watermark algorithm based on DWT-SVD by using the Y component of YCbCr color space from a key frame [13]. He embeds the same watermark into all video key frames. Experimental results show that this algorithm can't deal with attacks well, such as, frame dropping, frame averaging, and frame swapping. It's ability to respond to these attacks, Ramakrishnan et al. use the R component of RGB color space from video key frames to propose a more robust video watermark algorithm based

Fig. 12. Watermarked key frames of the 4th scene from suzie.avi to be tested, R components $\tilde{I}_{0,4}, \tilde{I}_{1,4}$ and $\tilde{I}_{2,4}$, *LH* wavelet component coefficients $\tilde{C}_{0,4}, \tilde{C}_{1,4}$ and $\tilde{C}_{2,4}$, target wavelet coefficients $\tilde{Q}_{0,4}, \tilde{Q}_{1,4}$ and $\tilde{Q}_{2,4}$, extracted watermarks $\bar{w}_{0,4}, \bar{w}_{1,4}$ and $\bar{w}_{2,4}$ from key frames, and the extracted watermark $\tilde{w}_4$ obtained by combining $\bar{w}_{0,4}$, $\bar{w}_{1,4}$ and $\bar{w}_{2,4}$ using (19).

on DWT-SVD. They embed different watermarks into video key frames from different video scenes [33]. However, they don't measure the effect of wavelet coefficients on video frame distortion, i.e., they believe that the video frame distortion caused by changing different wavelet coefficients is the same.

4.1.4. Attack methods

We mimic different types of video attacks to watermarked video, and measure the imperceptibility of watermark and robustness of watermarked video by the PSNR and NCC values. Adding noise in a video frame means adding some unnecessary or redundant information randomly in it. The common noise includes Gaussian noise, Salt and Pepper noise and Poisson noise. Gaussian noise means that the value of each pixel of the video frame changes, and the amplitude of the change follows a Gaussian distribution where the mean value is a and the variance is b. In this work, the Gaussian noise whose mean value is a and variance is b is denoted as $N(a, b)$. Salt and Pepper noise means randomly changing some pixel values in video frames to 0 or 255, such as changing some pixel values in the light area to 0 or changing the pixel values in the dark area to 255. We use $c\%$ to indicate that salt and pepper noise is added to $c\%$ pixels of a video frame. Poisson noise means that if a video frame has a pixel value of p ($p > 0$) at a certain location, it is replaced by the corresponding output pixel value sampled from Poisson distribution with a mean of p. If the video frame has a pixel value of 0 at a certain location, then the pixel value doesn't change.

Video frame blur is the convolution of a video frame with a Gaussian low pass filter. In general, the dimension of a Gaussian filter is 3×3 and its variance is d, which is simply denoted as $[3, 3, d]$.

Frame dropping means some of the video frames are randomly dropped or discarded in video sequences. Its parameter $f\%$ represents that $f\%$ of the watermarked video frames are attacked by frame dropping.

Table 4
The PSNR value of different algorithms without any attack using different sizes of watermarks.

size	[13]	[33]	ours	improvement
32 × 32	63.552	66.798	**72.267**	+8.19%
64 × 64	63.989	66.354	**68.470**	+3.19%
128 × 128	58.230	59.857	**60.957**	+1.83%

Fig. 13. The different sizes of watermarks, i.e., the sizes of (a), (b), and (c) are 32 × 32, 64 × 64, and 128 × 128, respectively.

Frame averaging is that the current frame of a video sequence is replaced by the average of the current frame and two nearest frames. Its parameter *f*% represents that *f*% of the watermarked video frames are attacked by frame averaging.

Frame swapping means to exchange the current frame of video with the other random video frame. Its parameter *f*% represents that *f*% of the watermarked video frames are attacked by frame swapping.

Frame scaling means the resizing of frames, and we use the default settings from the "imresize ()" function in the matlab toolbox.

Frame rotation means the rotation of a video frame in a certain direction, we use the default settings from the "imrotate ()" function in the matlab toolbox.

Frame cropping means to crop a block of pixels of a particular size from a video frame.

4.2. Imperceptibility experiment

In order to measure the imperceptibility for our algorithm, we test the quality of a watermarked video sequence without any attack and with different types of attacks, respectively.

4.2.1. Imperceptibility experiment without any attack

For the experiments of watermarked video without any attack, we embed the same size of the watermark into video key frames from each video scene, where video key frames from each video scene are shown in Table 2. The embeded watermarks with different sizes 32 × 32, 64 × 64 and 128 × 128 are shown in Fig. 13. These watermarks can be downloaded from https://www.qust.edu.cn/xywh/UIS/sjsbxt_VI_.htm. For the watermarks shown in Figs. 13(a)-(c), when they are embedded into the original video respectively, the NCC values of the extracted watermarks from watermarked video with no attack, first reach their upper limit 1.0 when $\alpha = 1.5$, $\alpha = 1.3$ and $\alpha = 1.5$, respectively.

As we know, generally, the bigger PSNR value, the better frame quality. The PSNR value of our algorithm is much bigger than those from the algorithms [13,33] according to Table 4. In other words, the original video quality hardly changes after the watermark is embedded via our proposed algorithm, because we choose the wavelet coefficients with minimum distortion cost to embed the watermark. But algorithms in [13,33] select all the wavelet coefficients which may incur large distortion cost. Specifically, when the size of embedded watermark is smaller, our algorithm has more obvious advantages compared with algorithm [13,33]. For example, when the size of the embedded watermarks are 32 × 32, 64 × 64 and 128 × 128, respectively, the PSNR values of our algorithm are 72.267, 68.470 and 60.957, respectively, and the percentage of improvement compared with the state-of-the-art algorithm [33] is 8.19%, 3.19% and 1.83%, respectively. In our algorithm, we need to select more wavelet coefficients when the size of embedded watermark is larger, the advantages of our algorithm are less and less significant over the state-of-the-art algorithms [13,33].

We also use the DSCQS method introduced in Section 3.1 to evaluate the quality of the watermarked video. The results show that the quality of the watermarked video based on our method and the state-of-the-art methods [13] and [33] are all excellent. As shown in Fig. 14, we show the watermarked video frames based on different algorithms. Specifically, Figs. 14(a)-(c) are the watermarked video frames by embedding Figs. 13(a)-(c) into the original 300th frame based on [13], respectively. Figs. 14(d)-(f) and Figs. 14(g)-(i) are based on [33] and our work, respectively. We can see that there is no obvious difference in the visual quality in each column. Therefore, all of the algorithms have excellent visual quality based on the observers'

Fig. 14. (a)-(c) are the watermarked video frames obtained by embedding Figs. 13(a)–(c) into the original 300th frame based on [13], respectively. (d)-(f) are the watermarked video frames based on [33], respectively. (g)-(i) are the watermarked video frames based on our algorithm, respectively.

Table 5
SSIM of different algorithms without any attack using different size watermarks.

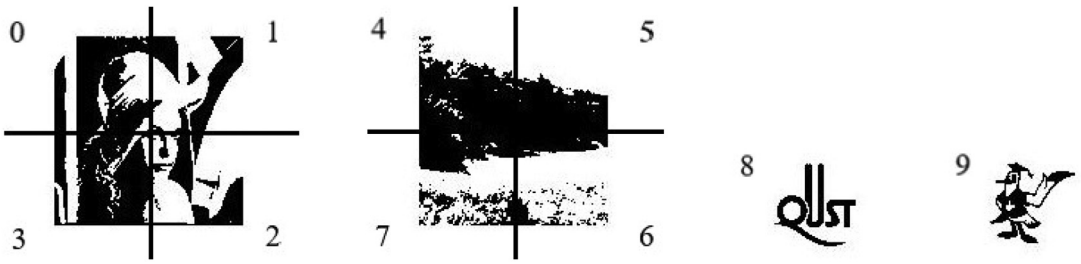
size	[13]	[33]	ours	improvement
32 × 32	0.999	0.999	**1.000**	+0.1%
64 × 64	0.998	0.998	**1.000**	+0.2%
128 × 128	0.995	0.997	**1.000**	+0.3%

result with the DSCQS method. This is expected because all the algorithms embed the watermark into singular values of target wavelet coefficients. Slight variation of singular values does not affect the visual perception of a video frame.

As there is no obvious difference under subjective evaluation by using DSCQS, we compare the SSIM (structural similarity) value of the watermarked video of our algorithm with those of the state-of-the-art methods in Table 5. We find that their SSIM values are all very close to the maximum value 1.000. The main reason is that they embed watermark into the singular value of SVD decomposition of wavelet coefficient and the change of wavelet coefficient after embedding watermark into a video frame is small. In other words, $2\mu_{I_l}\mu_{I_l^*} \approx \mu_{I_l}^2 + \mu_{I_l^*}^2$, $2\sigma_{I_l,I_l^*} \approx \sigma_{I_l}^2 + \sigma_{I_l^*}^2$. Therefore their SSIM values are all close to 1.000. We also notice that the SSIM value based on our method is slightly closer to 1.000 than its two peers [13,33]. The main reason is that we choose the wavelet coefficients with the minimum distortion cost to embed the watermark.

4.2.2. Imperceptibility experiment under different types of attacks

In Table 2, we divide Suzie.avi into 10 scenes, labeled as *Scene*0, *Scene*1, ⋯, and *Scene*9, respectively. We select 10 different watermarks whose dimension is 64 × 64, as shown in Fig. 15, labeled as 0, 1, ⋯, 9, corresponding to the 10 scenes.



(a) lena (b) scene (c) qust (d) mascot

Fig. 15. Candidate watermarks.

Table 6
The mean PSNR value of all the video key frames in different attack experiments, where the bigger PSNR, the better frame quality.

Attack Type	PSNR				
	Parameters	[13]	[33]	ours	compared
No attack		62.770	64.897	**73.582**	+13.38%
Gaussian	$N(0, 0.01)$	24.817	24.818	24.818	0.00%
	$N(0, 0.05)$	18.210	18.210	**18.211**	+0.01%
	$N(0, 0.1)$	15.919	15.919	15.919	0.00%
Salt and Pepper	1%	**30.462**	30.456	30.460	-0.01%
	5%	23.468	23.468	**23.470**	+0.01%
	10%	20.457	20.459	20.459	0.00%
Poisson	–	32.057	32.058	**32.059**	+0.01%
Blur	[3,3,0.1]	64.537	67.170	**71.305**	+6.16%
	[3,3,0.2]	64.538	67.171	**71.306**	+6.18%
	[3,3,0.3]	64.576	67.221	**71.370**	+6.17%
Frame dropping	10%	60.765	62.979	**66.840**	+6.14%
	20%	56.792	58.923	**61.561**	+4.48%
	25%	54.232	56.167	**59.690**	+6.27%
Frame averaging	10%	59.915	62.221	**66.346**	+6.63%
	20%	50.102	57.349	**60.711**	+5.86%
	25%	54.905	56.083	**57.668**	+5.03%
Frame swapping	10%	61.384	63.667	**67.911**	+6.13%
	20%	58.678	62.124	**63.917**	+4.57%
	25%	57.170	59.294	**62.345**	+5.15%
Scaling	–	**42.322**	42.320	42.321	+0.01%
Rotation	–	21.567	21.567	21.567	+0.00%
Cropping	–	29.690	29.687	**29.693**	+0.01%

Specifically, we embed the 0*th* watermark into all video key frames from *Scene*0, we embed the 1*st* watermark into all video key frames from *Scene*1 etc.

The suitable scaling factor for this example is $\alpha = 1.2$ because the NCC value of the extracted watermark from the watermarked video sequence with no attack first reaches its upper limit 1.0 when $\alpha = 1.2$. In this case, the PSNR value of our algorithm is 72.267 and the percentage of improvement compared with the state-of-the-art algorithm [33] is 13.38%, as shown in Table 6.

For imperceptibility experiment with different types of attacks, we select the attack types involved in the algorithm [33] to attack watermarked video, such as, Gaussian noise with parameters $N(0, 0.01)$, $N(0, 0.05)$, and $N(0, 0.1)$, Salt and Pepper with parameters 1%, 5% and 10%, Poisson noise, Frame dropping with parameters 10%, 20% and 25%, Frame averaging with parameters 10%, 20% and 25%, Frame swapping with parameters 10%, 20% and 25%, Scaling, Rotation and Cropping, as shown in Table 6.

We can see that the PSNR value of our algorithm after being attacked is greatly improved compared with [33] and [13] from Table 6, especially for Frame blur, Frame dropping, Frame averaging and Frame swapping. The main reason is that these attacks mainly affect the texture area of video frames and watermarked ones obtained by our algorithm are closer to the original video frames without watermark. For example, when the watermarked video frame is attacked by video frame blur, the PSNR of our algorithm increased by 6% over the state-of-the-art algorithm [33].

However, when a watermarked video sequence is attacked by adding noise, the PSNR values of [13,33] and our proposed one are almost identical. One reason is that we add noise at the same pixel position of a video frame, while the amplitude of pixel value modification is the same for different algorithms. Another reason is that a video frame has obvious visual difference after adding noise, that is to say, the video frame distortion caused by adding noise is far greater than the dis-

Table 7
The mean SSIM value of all the video key frames in different attack experiments.

Attack Type	SSIM				
	Parameters	[13]	[33]	ours	difference
No attack		0.999	0.999	**1.000**	+0.10%
Gaussian	$N(0, 0.01)$	0.179	0.179	0.179	0.00%
	$N(0, 0.05)$	0.061	0.061	0.061	+0.00%
	$N(0, 0.1)$	0.040	0.040	0.040	0.00%
Salt and Pepper	1%	0.718	0.719	0.719	0.00%
	5%	0.255	0.255	0.255	0.00%
	10%	0.118	0.118	0.118	0.00%
Poisson	–	0.508	0.508	0.508	0.00%
Blur	[3,3,0.1]	1.000	1.000	1.000	0.00%
	[3,3,0.2]	1.000	1.000	1.000	0.00%
	[3,3,0.3]	1.000	1.000	1.000	0.00%
Frame dropping	10%	0.989	0.988	**0.992**	+0.30%
	20%	0.975	0.976	**0.978**	+0.21%
	25%	0.971	0.972	**0.973**	+0.10%
Frame averaging	10%	0.982	0.982	**0.985**	+0.31%
	20%	0.965	0.960	**0.966**	+0.10%
	25%	0.956	0.954	0.956	0.00%
Frame swapping	10%	0.989	0.989	**0.992**	+0.30%
	20%	0.976	0.982	**0.984**	+0.21%
	25%	0.975	0.977	0.977	0.00%
Scaling	–	0.971	0.971	0.971	0.00%
Rotation	–	0.902	0.902	0.902	0.00%
Cropping	–	0.974	0.974	0.974	0.00%

tortion caused by embedding watermark into a video frame. Moreover, a video frame has obvious visual difference under geometric attacks, such as scaling, rotation and cropping, i.e., the video frame distortion caused by geometric attacks is far greater than the distortion caused by embedding watermark into a video frame. Thus the difference of watermarked video frames after adding noise and geometric attacks between the work [13,33] and ours is not obvious where the PSNR values are almost the same.

We present the SSIM values of the watermarked video frames under different types of attacks in Table 7. We find that the SSIM values of our method are almost the same as those of [13] and [33].

Moreover, we use DSCQS to evaluate the subjective quality of the watermarked video under different types of attacks. According to the observers' results, we find that the subjective quality of the watermarked video is almost the same between the work [13], [33] and ours. For example, we show the 300th frame in watermarked video based on [13], [33] and our work under Gaussian noise, frame averaging and frame rotation attacks in Fig. 16. Specifically, Figs. 16(a)-(c) are the watermarked video frames based on [13] under Guassian noise, frame averaging and frame rotation attacks, respectively. Figs. 16(d)-(f) and Figs. 16(g)-(i) are the watermarked video frames based on [33] and our work, respectively. We have the same poor visual quality based on the observer' results for Gaussian noise from them. We also have the same excellent and medium visual quality for frame averaging and frame rotation attacks from them. The main reason is that the frame distortion caused by these attacks is much greater than that caused by embedding a watermark into a frame.

4.3. Robustness Experiment

In this part, we use NCC values to measure the algorithm robustness under the imperceptibility experiment with different types of attacks. For watermarked video with no attack, the NCC values of three tested algorithms are all 1. But the NCC value is less than 1 if the watermarked video is attacked. After watermarked video is attacked by one type, the lager NCC, the better the visual quality of the extracted watermark, i.e., the better the robustness of a watermark algorithm against this attack type.

We can see that most of NCC values in our algorithm increase by 10% on average over its two peers when the watermarked video is added with noise from Table 8. We choose wavelet coefficients with minimum distortion cost to embed the watermark. When video frames are attacked by adding noise, the changes of these wavelet coefficients are relatively small. That is to say, our algorithm is more likely to extract the correct information. There are more watermark pixel coordinates (i, j) in our algorithm such that $w(i, j) \times w^*(i, j) = 1$ in (10) than its peers, where $i \in \mathbb{N}_p$, $j \in \mathbb{N}_q$. Therefore, when a watermarked video frame is attacked by adding noise, our algorithm has a higher NCC value than its peers [13,33].

In addition, when the watermarked video is attacked by certain attack types, such as frame blur, frame dropping, frame averaging, frame swapping or geometric attacks, our algorithm is only comparable with its two peers. In some special cases, e.g., frame dropping with parameters 10% and 20%, our algorithm has slightly worse NCC values. The main reason is as

Fig. 16. (a)-(c) are the 300th frame in the watermarked video based on [13] under Gaussian noise, frame averaging and rotation, respectively. (d)-(f) and)(g)–(i) are the frames based on [33] and our work, respectively.

follows. These attack types can cause the distortion of the texture area of a video frame. NCC values with the algorithms [13,33] and ours all decrease sharply because all of us make use of the *LH* component from a video frame to embed the watermark, where the *LH* component is the horizontal texture area of a video frame. Moreover, the algorithms [13,33] apply SVD to all *LH* component coefficients and embed the watermark into the large singular value from the singular value matrix. However, because our method applies SVD to some *LH* component coefficients with low distortion cost, the singular values embedded in our algorithm are smaller than those embedded in algorithms [13,33]. When the video frame is attacked by frame blur, frame dropping, frame averaging, or frame swapping, the smaller singular value is more likely to be affected because it corresponds to the texture area of a video frame. When a video frame is attacked by scaling, the wavelet coefficients of watermarked video frames are quite different from those of the original watermarked ones because DWT lacks shift invariance [20]. In other words, almost all of watermark pixel coordinates (i, j) are such that $w(i, j) \times w^*(i, j) = 0$ in (10), where $i \in \mathbb{N}_p$, $j \in \mathbb{N}_q$. Hence, the values of NCC are close to 0 under scaling. The NCC value of our algorithm under rotation is slightly higher than those of algorithms [13,33] because we apply SVD to those wavelet coefficients with low distortion cost and those wavelet coefficients are concentrated in the frame texture area. Most of those coefficients are restored when we reversely rotate the frame, i.e., there are more watermark pixel coordinates in ours algorithm such that $w(i, j) \times w^*(i, j) = 1$ in (10). When a video frame is attacked with cropping, the values of NCC are close to 1 because the size of the cropping block is relatively small and we integrate multiple similar watermarks from key frames in the scene using (19).

Based on the results of imperceptibility and robustness experiments shown above, we can see that our proposed algorithm has better imperceptibility and robustness than the state-of-the-art algorithms [13,33].

Table 8
NCC value of all the video key frames under different attacks.

Attack Type	NCC				
	Parameters	[13]	[33]	ours	difference
No attack		1.000	1.000	1.000	0.00%
Gaussian	$N(0, 0.01)$	0.881	0.907	**0.992**	+9.37%
	$N(0, 0.05)$	0.921	0.933	**0.999**	+7.07%
	$N(0, 0.1)$	0.929	0.937	**1.000**	+6.72%
Salt and Pepper	1%	0.845	0.878	**0.990**	+12.76%
	5%	0.895	0.918	**0.998**	+8.72%
	10%	0.913	0.928	**0.999**	+7.65%
Poisson	–	0.826	0.855	**0.980**	+14.62%
Blur	[3,3,0.1]	1.000	1.000	1.000	0.00%
	[3,3,0.2]	1.000	1.000	1.000	0.00%
	[3,3,0.3]	1.000	1.000	1.000	0.00%
Frame dropping	10%	**0.980**	0.978	0.971	-0.92%
	20%	**0.923**	0.909	0.918	-0.54%
	25%	0.861	0.850	**0.882**	+2.44%
Frame averaging	10%	0.976	0.975	**0.976**	0.00%
	20%	**0.919**	0.914	0.912	-0.76%
	25%	0.843	0.869	**0.874**	+0.58%
Frame swapping	10%	0.985	**0.987**	0.977	-1.01%
	20%	0.938	**0.947**	0.931	-1.69%
	25%	0.843	0.869	**0.874**	+0.58%
Scaling	–	0.003	0.003	0.003	0.00%
Rotation	–	0.896	0.918	**0.935**	+1.85%
Cropping	–	0.999	0.999	0.999	0.00%

5. Conclusion

Current video watermark algorithms based on DWT-SVD [8,13,18,25–33] suffer from poor imperceptibility and robustness under attack because they either embed the same watermark into all the key frames or assume the wavelet coefficients have equal distortion cost.

In this paper, we divide the original video sequence into several scenes based on the histogram differences among adjacent frames and we embed different watermarks into different scenes. Next, we design a method to measure the video frame distortion caused by changing different wavelet coefficients. Then, we conclude the watermark embedding problem in one video scene as an optimization problem to minimize video frame distortion cost with respect to the size of the watermark. Finally, we compare our method with two state-of-the-art methods [13,33]. In conclusion, experimental results show that the PSNR value of our proposed algorithm is significantly improved over the prior arts in [13,33], and the NCC value is better than them when under various noise attacks.

References

[1] A. Piva, F. Bartolini, M. Barni, Managing copyright in open networks, IEEE Internet Computing 6 (3) (2002) 18–26.
[2] G. Doerr, J.-L. Dugelay, A guide tour of video watermarking, Signal Processing: Image Communication 18 (4) (2003) 263–282.
[3] A. Joseph, K. Anusudha, Robust watermarking based on DWT-SVD, International Journal on Signal & Image Security 1 (1) (2013) 1–5.
[4] C.I. Podilchuk, E.J. Delp, Digital watermarking: algorithms and applications, IEEE Signal Processing Magazine 18 (4) (2001) 33–46.
[5] M.U. Celik, G. Sharma, A.M. Tekalp, E. Saber, Lossless generalized-LSB data embedding, IEEE Transactions on Image Processing 14 (2) (2005) 253–266.
[6] P. Bas, J.-M. Chassery, B. Macq, Geometrically invariant watermarking using feature points, IEEE Transactions on Image Processing 11 (9) (2002) 1014–1028.
[7] C.-W. Tang, H.-M. Hang, A feature-based robust digital image watermarking scheme, IEEE Transactions on Signal Processing 51 (4) (2003) 950–959.
[8] P.W. Chan, M.R. Lyu, R.T. Chin, A novel scheme for hybrid digital video watermarking: approach, evaluation and experimentation, IEEE Transactions on Circuits and Systems for Video Technology 15 (12) (2005) 1638–1649.
[9] H.-Y. Huang, C.-H. Yang, W.-H. Hsu, A video watermarking technique based on pseudo-3-D DCT and quantization index modulation, IEEE Transactions on Information Forensics and Security 5 (4) (2010) 625–637.
[10] M.-J. Lee, D.-H. Im, H.-Y. Lee, K.-S. Kim, H.-K. Lee, Real-time video watermarking system on the compressed domain for high-definition video contents: Practical issues, Digital Signal Processing 22 (1) (2012) 190–198.
[11] N.M. Makbol, B.E. Khoo, Robust blind image watermarking scheme based on redundant discrete wavelet transform and singular value decomposition, AEU-International Journal of Electronics and Communications 67 (2) (2013) 102–112.
[12] P. Bao, X. Ma, Image adaptive watermarking using wavelet domain singular value decomposition, IEEE Transactions on Circuits and Systems for Video Technology 15 (1) (2005) 96–102.
[13] O.S. Faragallah, Efficient video watermarking based on singular value decomposition in the discrete wavelet transform domain, AEU-International Journal of Electronics and Communications 67 (3) (2013) 189–196.
[14] M. Barni, F. Bartolini, V. Cappellini, A. Piva, A DCT-domain system for robust image watermarking, Signal Processing 66 (3) (1998) 357–372.
[15] Z.-M. Lu, H.-Y. Zheng, J.-W. Huang, A digital watermarking scheme based on DCT and SVD, in: Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007), volume 1, IEEE, 2007, pp. 241–244.
[16] S.D. Roy, X. Li, Y. Shoshan, A. Fish, O. Yadid-Pecht, Hardware implementation of a digital watermarking system for video authentication, IEEE Transactions on Circuits and Systems for Video Technology 23 (2) (2012) 289–301.
[17] H. Flórez, M. Argáez, A model-order reduction method based on wavelets and POD to solve nonlinear transient and steady-state continuation problems, Applied Mathematical Modelling 53 (2018) 12–31.

[18] C.-C. Lai, C.-C. Tsai, Digital image watermarking using discrete wavelet transform and singular value decomposition, IEEE Transactions on Instrumentation and Measurement 59 (11) (2010) 3060–3063.
[19] N. Kingsbury, A dual-tree complex wavelet transform with improved orthogonality and symmetry properties, in: Proceedings 2000 International Conference on Image Processing (Cat. No. 00CH37101), volume 2, IEEE, 2000, pp. 375–378.
[20] M. Asikuzzaman, M.R. Pickering, An overview of digital video watermarking, IEEE Transactions on Circuits and Systems for Video Technology 28 (9) (2017) 2131–2153.
[21] A.A. Yahia, E.P. Del Barrio, Thermal systems modelling via singular value decomposition: direct and modular approach, Applied Mathematical Modelling 23 (6) (1999) 447–468.
[22] Y.L. Chen, J. Wen, Inverse estimation of indoor airflow patterns using singular value decomposition, Applied Mathematical Modelling 36 (6) (2012) 2627–2641.
[23] H.A. Abdallah, M.M. Hadhoud, A.A. Shaalan, Svd-based watermarking scheme in complex wavelet domain for color video, in: 2009 International Conference on Computer Engineering & Systems, IEEE, 2009, pp. 455–460.
[24] M. Asikuzzaman, M.J. Alam, M.R. Pickering, A blind and robust video watermarking scheme in the DT CWT and SVD domain, in: 2015 Picture Coding Symposium (PCS), IEEE, 2015, pp. 277–281.
[25] M. Ejima, A. Miyazaki, A wavelet-based watermarking for digital images and video, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 83 (3) (2000) 532–540.
[26] K.R. Chetan, K. Raghavendra, DWT based blind digital video watermarking scheme for video authentication, International Journal of Computer Applications 4 (10) (2010) 19–26.
[27] A. Naved, Y. Rajesh, Dual band watermarking using 2-D DWT and 2-level SVD for robust watermarking in video, Int. J. Sci. Res.(IJSR), India Online ISSN (2013) 2319–7064.
[28] L. Rajab, T. Al-Khatib, A. Al-Haj, Hybrid DWT-SVD video watermarking, in: 2008 International Conference on Innovations in Information Technology, IEEE, 2008, pp. 588–592.
[29] R.T. Paul, Review of robust video watermarking techniques, IJCA Special Issue on Computational Science 3 (2011) 90–95.
[30] G. Bhatnagar, B. Raman, A new robust reference watermarking scheme based on DWT-SVD, Computer Standards & Interfaces 31 (5) (2009) 1002–1013.
[31] M. Shanmugam, A. Chokkalingam, Performance analysis of 2 level DWT-SVD based non blind and blind video watermarking using range conversion method, Microsystem Technologies 24 (12) (2018) 4757–4765.
[32] D. Singh, S.K. Singh, DWT-SVD and DCT based robust and blind watermarking scheme for copyright protection, Multimedia Tools and Applications 76 (11) (2017) 13001–13024.
[33] S.P. alias Sathya, S. Ramakrishnan, Fibonacci based key frame selection and scrambling for video watermarking in DWT-SVD domain, Wireless Personal Communications (2018) 1–21.
[34] C.W.H. Fung, W. Godoy Jr, A new approach of DWT-SVD video watermarking, in: 2011 Third International Conference on Computational Intelligence, Modelling & Simulation, IEEE, 2011, pp. 233–236.
[35] D.K. Thind, S. Jindal, A semi blind DWT-SVD video watermarking, Procedia Computer Science 46 (2015) 1661–1667.
[36] T. Zong, Y. Xiang, I. Natgunanathan, S. Guo, W. Zhou, G. Beliakov, Robust histogram shape-based method for image watermarking, IEEE Transactions on Circuits and Systems for Video Technology 25 (5) (2015) 717–729.
[37] M. Andalibi, D.M. Chandler, Digital image watermarking via adaptive logo texturization, IEEE Transactions on Image Processing 24 (12) (2015) 5060–5073.
[38] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Transactions on Image Processing 13 (4) (2004) 600–612.
[39] R.I.-R. BT, Methodology for the subjective assessment of the quality of television pictures, International Telecommunication Union (2002).
[40] S. Gholizadeh, O.A. Samavati, Structural optimization by wavelet transforms and neural networks, Applied Mathematical Modelling 35 (2) (2011) 915–929.
[41] M. Al Wadia, M. Tahir Ismail, Selecting wavelet transforms model in forecasting financial time series data based on ARIMA model, Applied Mathematical Sciences 5 (7) (2011) 315–326.
[42] S. Mallat, A Wavelet Tour of Signal Processing, Elsevier, 1999.
[43] V. Holub, J. Fridrich, T. Denemark, Universal distortion function for steganography in an arbitrary domain, EURASIP Journal on Information Security 2014 (1) (2014) 1.
[44] N. Otsu, A threshold selection method from gray-level histograms, IEEE Transactions on Systems, Man, and Cybernetics 9 (1) (1979) 62–66.