Multimed Tools Appl (2018) 77:663–688
DOI 10.1007/s11042-016-4268-3

# Noise-level estimation based detection of motion-compensated frame interpolation in video sequences

Ran Li[1,2] · Zhenghui Liu[1,3] · Yu Zhang[1] · Yanling Li[1] ·
Zhangjie Fu[2]

Received: 8 June 2016 / Revised: 11 December 2016 / Accepted: 13 December 2016 /
Published online: 30 December 2016

**Abstract** Motion-Compensated Frame Interpolation (MCFI) is commonly used to produce the fake high-frame-rate videos, and it can be regarded as a video forgery operation from a broad sense. In this paper, we use the noise-level estimation to expose MCFI operator, and exploit the periodicity of noise-level varying to propose an effective automatic detection method. To guarantee the high detection accuracy, the high-pass filtering and the spike enhancement are both employed to extract the peak outliers in the Fourier domain. Depending on these outliers, we design the criterion of credibility value to make a final decision. The extensive experiments evaluated on hundreds of video sequences with different spatial resolutions and two parameter configurations of H.264/AVC have shown that the validity of the proposed method, which has the better detection accuracy for the MCFI method and the frame repetition.

**Key words** Video forensics · Motion-compensated frame interpolation · Noise-level estimation · Spectral analysis

✉ Ran Li
liran358@163.com

[1] School of Computer and Information Technology, Xinyang Normal University, Nanhu Road #237, Xinyang 464000, China

[2] School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing 210003, China

[3] College of Information Engineering and Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China

## 1 Introduction

Video tampering is becoming much easier with rapid development of various video editing tools such as VideoEdit Magic, therefore the forensics techniques are required to verify the authenticity and integrity of digital video [9, 19, 22].

Digital video is the sequence of still images along the temporal dimension, and its specific forgery is the frame-based manipulation. Until now, many research works has always focused on the blind video inter-frame forensics of frame duplication, frame deletion and frame adding [12]. The current frame-based tampering is only used to cover the some key video clips, however, it is a special frame-adding operation that Frame Rate Up-Conversion (FRUC) [1, 2, 15] to insert periodically several new frames into the video sequence, which is proposed to improve the visual quality of low frame-rate video. Forgers often use the FRUC to generate the faked high-frame-rate videos, especially for videos over Internet. Therefore, the reports about detecting FRUC are increasing in recent years. To the best of our knowledge, the pioneering work was proposed by Bian et al. [4] to detect the video FRUC by using the periodic properties of inter-frame similarity. However, their method can only effectively identify the fake high-frame-rate video by frame repetition. Moreover, this work was further extended to investigate the specific artifacts of those fake bitrate videos [5]. In the research field of FRUC, it is well known that the one of the simplest techniques is the combination of adjacent video frames, like that frame repetition or frame averaging. Although they have a good viewing experience for static regions, the jittering or ghosting artifacts often occur because the motion between successive frames is neglected. To improve the visual quality of video, the forgers are more likely to perform the Motion-Compensated Frame Interpolation (MCFI) [16, 17], which exploits the motion trajectories to interpolate the new frames. Therefore, we need some advanced forensic techniques to defeat those fake high-frame-rate videos produced by MCFI. Bestagini et al. [3] is firstly trying to detect the MCFI, and their idea is derived from the discovery that the periodicity exists in the prediction error between the forged video and its re-interpolated estimator. Their technique cannot offer an automatic recognition system, and besides the re-interpolation, which is the core of detection algorithm in [3], also introduces excessive computations. By the experience that MCFI leads to edge discontinuity or over-smoothing artifacts around object boundaries, Yao et al. [20] proposed to measure the edge-intensity for the detection of MCFI operation. Compared with the approach of [3], their work adds the automatic recognition by exploiting the Kaufman adaptive moving average that defines an adaptive threshold to distinguish the interpolated frames by MCFI from the original frames. The temporal variation of edge-intensity is the external phenomenon resulting from the lack of the high-frequency components in the interpolated frames. However, the existing advanced MCFI methods can recover accurately the details of edge and texture for some video sequences with few high-frequency components, which affects the accuracy of MCFI forensics method in [20]. Therefore, instead of detecting the blur effects, we require more some traces left by the key operation in MCFI to verify the forgery. In MCFI, the interpolated frames are obtained by block-based average along with the motion trajectories. Considering that the noise inevitably exists in the video sequence, the averaging of similar pixel values can weaken some noise level in the interpolated frame. Since the interpolated frames are periodically inserted into the original frames, we can detect the MCFI operator by exploiting the temporal periodicity of noise levels in video frames.

In this paper, we propose a new video MCFI forensic method to analyze the detectable effects introduced by an element of MCFI. The main contributions of this work can be summarized as follows:

- We propose to expose MCFI operator by revealing the specific noise-level variation along the time dimension of video sequence, and this effect results from the noise accumulation of averaging the pixel values in the interpolation process.
- We use the noise-level variation to develop an automatic video MCFI detection method. After a series of specialized Fourier spectrum processing, a robust hard-threshold operation is exploited to make a decision. In addition, the proposed method is also suitable for the frame repetition by adding a pre-processing operation.

The video database with a moderate capacity is constructed to test the accuracy of our MCFI detection method, and the experimental results verify the high detection accuracy for various MCFI methods.

## 2 Detectable effects in MCFI

### 2.1 MCFI overview

The basic elements of MCFI include the Motion Estimation (ME), Motion Vector Smoothing (MVS) and Motion-Compensated Interpolation (MCI), and they form a flow to generate the intermediate frame $\boldsymbol{f}_{t.}$ between the previous frame $\boldsymbol{f}_{t.-1}$ and the following frame $\boldsymbol{f}_{t.+1}$ as shown in Fig. 1. Based on the translational motion model, the ME performs the block matching algorithm with different search schemes (e.g., full search, 3DRS search [10], etc.) between $\boldsymbol{f}_{t.-1}$ and $\boldsymbol{f}_{t.+1}$ to compute the motion vector of each block in the intermediate frame $\boldsymbol{f}_{t.}$, then the MVS corrects the outliers existing in the estimated motion vectors along the spatio-temporal direction. Finally, the MCI predicts the intermediate frame $\boldsymbol{f}_{t.}$ in terms of the motion vectors estimated by ME and MVS. Due to the non-stationary of video signal, it is not easy to accurately estimate the motions of objects, therefore the most research works are trying to improve the performance of ME and MVS, e.g., Dikbas et al. [7] selects the different predictors of motion vectors to impose implicit smoothness constraints into the block-matching algorithm, Yoo et al. [21] smoothes and refines both the forward and backward motion vectors, and selects the reliable one as the final result. Even with various ME and MVS strategies, the computing method of MCI in MCFI has few changes. Let $f_{t.}(x, y)$, $f_{t.-1}(x, y)$ and $f_{t.+1}(x, y)$ denote the pixels in the intermediate frame, the previous and following frame located at the spatial location $(x, y)$, respectively, and the formulation of MCI can be presented as

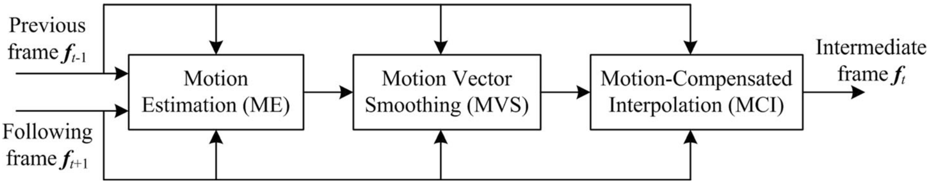

Previous frame $\boldsymbol{f}_{t-1}$
Following frame $\boldsymbol{f}_{t+1}$
Motion Estimation (ME)
Motion Vector Smoothing (MVS)
Motion-Compensated Interpolation (MCI)
Intermediate frame $\boldsymbol{f}_t$

**Fig. 1** The basic framework of MCFI

$$f_t(x,y) = \alpha_1 \cdot f_{t\text{-}1}\left(x + v_x, y + v_y\right) + \alpha_2 \cdot f_{t+1}\left(x - u_x, y - u_y\right), \tag{1}$$

where ($v_x$, $v_y$) and ($u_x$, $u_y$) represent the motion vectors pointing to the previous and following frames in the horizontal and vertical directions, $\alpha_1$ and $\alpha_2$ are the weighting coefficients inversely proportional to the distance from the interpolated frame to the original one, and their values are less than 1 but their sum equals 1, e.g., when the up-sampling factor is 2, $\alpha_1$ and $\alpha_2$ are both set to be 0.5. From Eq. (1), we can see that the each pixel of interpolated frame is computed by averaging the pixel values in the adjacent frames. However, the use of non-overlapped block in MCI usually leads to blocking artifacts at block boundaries. To reduce such artifacts, the Overlapped Block Motion Compensation (OBMC) and some adaptive variants [6] are commonly introduced into MCFI, but these schemes cannot reform the linear inherence of Eq. (1). Recently, it appears some works to present the non-linear MCI based on the Multiple-Hypotheses ME (MHME) [11, 13]. Firstly, they uses the traditional MCI to produce the multiple hypotheses of interpolated frame under the motion vector fields with different block sizes, then the intermediate frame is combined by these hypotheses with maximum a posterior probability according to the Bayesian interference. Based on the MCI method of [13], the work of [11] further optimizes the interpolated results by using some post-processing, e.g., texture optimization. By the summary on MCI methods, we can see that the averaging of pixel values from Eq. (1) is a necessary operator in the various MCFI methods, which give us an inspiration of exposing MCFI operation.

## 2.2 Gaussian noise accumulation in MCI

Because of the external circumstances and internal camera settings, the noise inevitably exists in a video sequence all the time. As a consequence of the Central Limit Theorem [18] for a large pixel number, the components of noise are commonly modeled as the a zero-mean additive Gaussian distribution, i.e., each pixel value in the previous frame $\boldsymbol{f}_{t\text{-}1}$ and the following frame $\boldsymbol{f}_{t+1}$ located at spatial location ($x$, $y$) can be represented as follows:

$$f_{t-1}(x,y) = o_{t-1}(x,y) + n_{t-1}(x,y), \tag{2}$$

$$f_{t+1}(x,y) = o_{t+1}(x,y) + n_{t+1}(x,y), \tag{3}$$

where $\boldsymbol{o}_{t\text{-}1}$ and $\boldsymbol{o}_{t+1}$ are respectively corresponding to the versions of $\boldsymbol{f}_{t\text{-}1}$ and $\boldsymbol{f}_{t+1}$ without the Gaussian noise, the $n_{t\text{-}1}(x, y)$ and $n_{t+1}(x, y)$ independently obey the zero-mean Gaussian distribution with unknown variance $\sigma^2$. Given the Eqs. (2) and (3), the Eq. (1) can be transformed as

$$\begin{aligned} f_t(x,y) &= \alpha_1 \cdot o_{t-1}\left(x + v_x, y + v_y\right) + \alpha_2 \cdot o_{t+1}\left(x - u_x, y - u_y\right) \\ &\quad + \alpha_1 \cdot n_{t-1}\left(x + v_x, y + v_y\right) + \alpha_2 \cdot n_{t+1}\left(x - u_x, y - u_y\right) \\ &= o_t(x,y) + n_t(x,y) \end{aligned} \tag{4}$$

with

$$o_t(x,y) = \alpha_1 \cdot o_{t-1}\left(x + v_x, y + v_y\right) + \alpha_2 \cdot o_{t+1}\left(x - u_x, y - u_y\right), \tag{5}$$

$$n_t(x,y) = \alpha_1 \cdot n_{t-1}\left(x + v_x, y + v_y\right) + \alpha_2 \cdot n_{t+1}\left(x - u_x, y - u_y\right), \tag{6}$$

where $\boldsymbol{o}_t$ is the noise-free version of $\boldsymbol{f}_{t,}$ $\boldsymbol{n}_t$ is still a zero-mean additive Gaussian distribution. We can derive the variance as

$$\sigma_t^2 = D[n_t(x,y)] = \left(\alpha_1^2 + \alpha_2^2\right)\sigma^2, \tag{7}$$

By Eq. (7), we can see that the MCI operator makes the variance of the interpolated frame be $\alpha_1{}^2 + \alpha_2{}^2$ (< 1) times than one of its reference frame, e.g., when the up-sampling factor is 2, $\sigma_t{}^2 = 0.5\sigma^2$. In fact, the MCI can be regard as a special image averaging. It is well known to average multiple exposures for static scenes to reduce noise variance. The pixel values of each block are approximately constant along the motion trajectory, which is similar to the multiple exposures for the same object. Therefore, the noise level is an important clue to reveal the MCI-based interpolation operator. Based on this property, the noise level of each frame can be used to detect the possible MCFI operator. We observe the varying pattern of noise levels in video sequence to distinguish video forged by MCFI from the originals.

## 2.3 Noise-level estimation

Considering that the high dimensionality of video signal, we employ a simple and fast wavelet-based technique, and it uses the Median Absolute Deviation (MAD) to estimate the Gaussian noise level of each frame, which depends on the assumption that the MAD of the fine-scale wavelet coefficients of image is proportional to the noise standard deviation [8]. Suppose a noisy $t$-th frame $\boldsymbol{f}_{t.}$ with additive Gaussian noise of the zero mean and unknown variance $\sigma_t{}^2$, and its size is $I_\mathrm{r} \times I_\mathrm{c}$ with $L = I_\mathrm{r} \times I_\mathrm{c}$ pixels in total. We firstly decompose $\boldsymbol{f}_{t.}$ by one-level fast discrete wavelet transform to obtain the fine-scale coefficients, and estimate the noise standard deviation as

$$\hat{\sigma}_t = \frac{1}{0.6745} \times \mathrm{MAD}(\mathbf{y}_t), \tag{8}$$

in which $\boldsymbol{y}_t$ is a column vector composed by the fine-scale coefficients. The MAD operation, for any column vector $\boldsymbol{x}$, is defined as the median of the absolute deviations from the median of the vector, i.e.,

$$\mathrm{MAD}(\mathbf{x}) = \mathrm{Median}(|\mathbf{x}{-}\mathrm{Median}(\mathbf{x})|), \tag{9}$$

where Median (·) is a filter to get the median element from the input vector. The fast discrete wavelet transform is realized by Mallat algorithm [14], its filterbank implementation takes only O($L$) operations. The MAD requires only O($L$log$_2$($L$)) operators. Therefore, due to a low time complexity, the wavelet-based technique is well-suited to estimating the noise levels of video frames in batches.

We select the 4-order symlets to compute the fine-scale wavelet coefficients. The five 512 × 512 test images *Lenna*, *Barbara*, *Peppers*, *Goldhill* and *Mandrill* are used to verify the validity of the MAD noise estimation method. The true noise levels of $\sigma$, of which range is from 5 to 200 by step 5, are used to contaminate the above test images, and then the average estimated noise levels $\hat{\sigma}$ of all noisy test images are derived by Eq. (8). Figure 2 shows the relation between the true $\sigma$ and the estimated $\hat{\sigma}$. It can be seen that the fitted curve by the real points ($\sigma$, $\hat{\sigma}$) is close to the ideal curve $\hat{\sigma} = \sigma$, i.e., the estimated $\hat{\sigma}$ is nearly same with the true $\sigma$. The average mean square error between the true $\sigma$ and estimated $\hat{\sigma}$ is only 0.3232, which proves that the MAD-based noise estimation has a better performance.

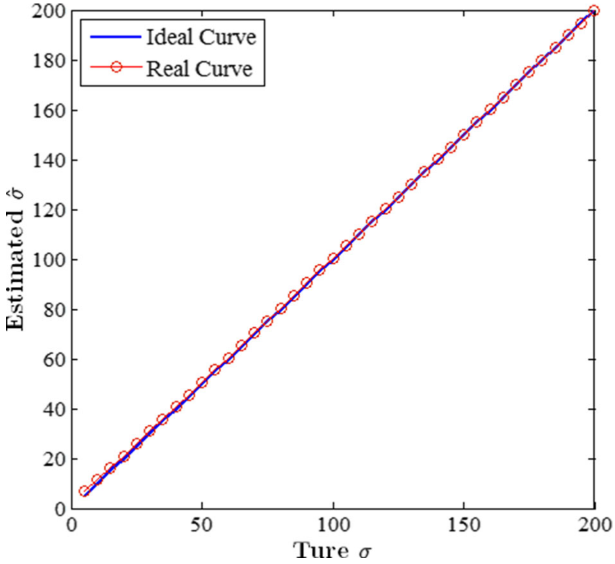


**Fig. 2** Relation between the true $\sigma$ and estimated $\hat{\sigma}$. The ideal curve represents $\hat{\sigma} = \sigma$, and the real curve represents the actual mapping between $\sigma$ and $\hat{\sigma}$

## 3 Detecting MCFI

As described previously, several new frames will be inserted into the resulting video after MCFI. It is expected that the noise level of the inserted frame will be smaller than that of its neighbors, because the averaging of pixel values in the MCI operator makes the variance of the reference frame be larger than the one of the interpolated frame. Due to the fact that these inserted frames are presented periodically, the key issue of the detection method is to determine whether there exists periodicity or not for those smaller noise levels in a suspected video clip. Therefore, as shown in Fig. 3, the flow of our method consists of three stages: (1) to compute the noise-level curve of suspected video; (2) to analyze the Fourier spectrum of noise-level curve, and compute the credibility value by the high-pass filtering, spike enhancement and extraction of peak outliers; (3) to set the threshold to distinguish the original video and those up-converted video, i.e., if the credibility value is greater than this threshold, the video is classified as a forged one, vice versa. More detailed descriptions on the method of detecting MCFI and the time-complexity analysis are presented in the following subsections.

### 3.1 Computation of noise-level curve

In our work, we firstly perform the 4-order symlets based fast discrete wavelet transform to obtain the fine-scale coefficients of each video frame at the first level, and then compute the corresponding standard deviation of noise by using Eq. (8) and (9). The set of $\{\sigma(t)|t = 1,2,\ldots,N\}$ is used to denote the noise-level curve of a given video, where $\sigma(t)$ is the standard deviation of the $t$-th frame $\boldsymbol{f}_{t.}$ in video and $N$ is the total frame number. If the $t$-th frame is an interpolated frame by MCFI, the corresponding $\sigma_t$ is expected to be smaller than those of the adjacent frames. Moreover, it is also observed that such smaller values would occur periodically.

Figure 4 illustrates the noise-level curve for both original and up-converted videos, in which the original video clip is the raw YUV sequence *Football* with CIF format and 30 fps.

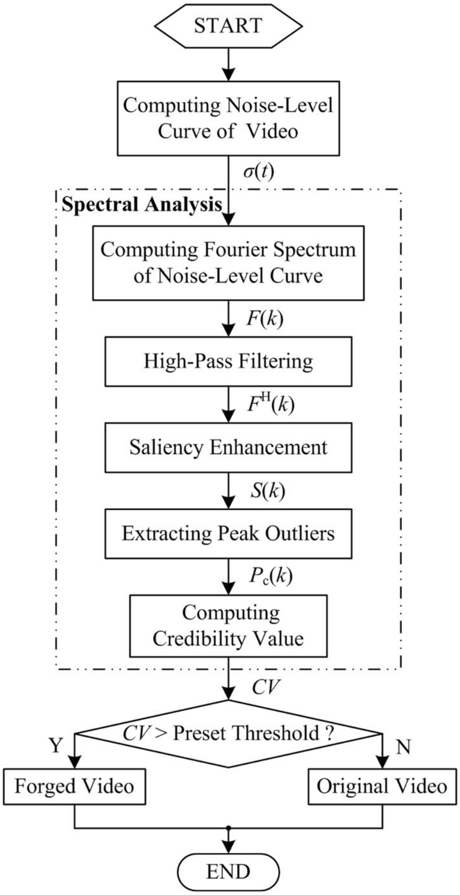


**Fig. 3** The flow chart of the proposed detecting method

For the up-converted video clip, we firstly down-sample the raw YUV sequence from 30 fps to 7.5 fps, and then up-converted it into 30 fps again by using the MCFI method in [21]. It can be seen that the original sequence has a smoothly varying noise level, but the interpolated sequence shows the periodic artifact, which is detectable in the Fourier domain.

### 3.2 Spectral analysis of noise-level curve

Considering that the periodic artifact is easy to be highlighted in the Fourier domain, we analyze the Fourier spectrum of noise-level curve to measure the periodicity.

Firstly, we transform the noise-level curve into frequency domain by using Fast Fourier Transform (FFT), and get the normalized frequency spectrums $F(k)$. As shown in Fig. 5a, e. It can be seen the spikes in the Fourier domain occurs only in the low-frequency range of original video, but the up-converted video adds the three new spikes in the medium and high frequency range. The spikes in the low-frequency range result from the smooth components of noise-level curve, and the spikes in the medium and high frequency range implies the periodic variance changing of inserted frames. Therefore, the high-pass filtering is then performed to eliminate the low frequency components, i.e.,

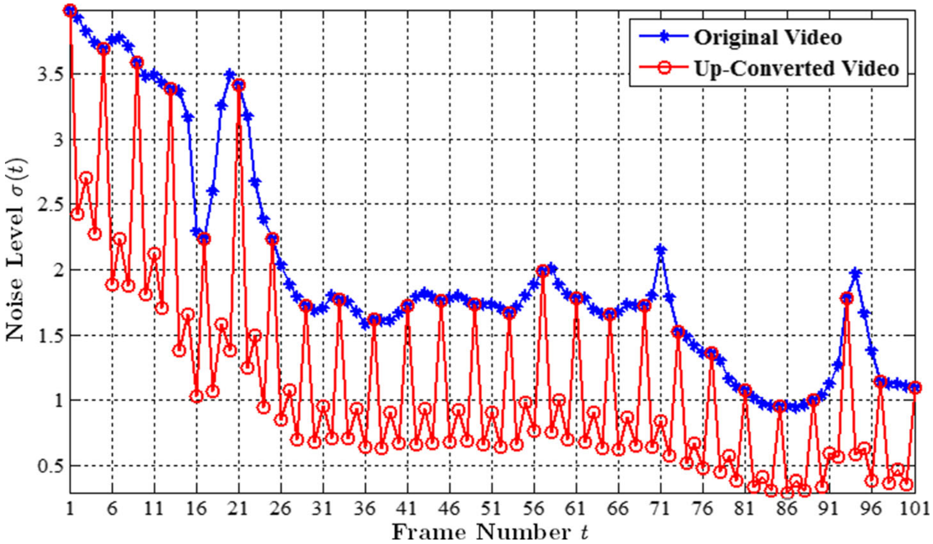
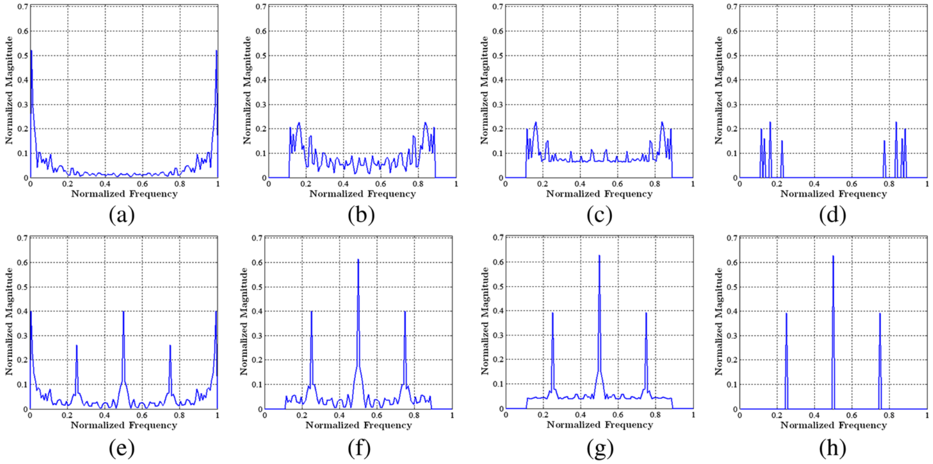


**Fig. 4** Illustrations of noise-level curves for both original and up-converted *Football* videos. The blue curve represents the original video at 30 fps without up-conversion; and the red curve represents the up-converted video from 7.5 fps to 30 fps. Note that we use the MCFI method proposed by [21]

$$F^{\mathrm{H}}(k) = \mathrm{HFP}[F(k), d] = \begin{cases} F(k), d < k \le N-d \\ 0 \quad , 0 \le k \le d \text{ or } N-d < k \le N \end{cases}, \tag{10}$$

where $F^{\mathrm{H}}(k)$ denotes the high-frequency coefficients of $F(k)$, and $d$ is the cut-off frequency. The Fourier spectrums of original and up-converted videos after high-pass filtering are presented in the Fig. 5b, f respectively. We can observe that no spike occurs in the Fourier domain of original video, and the up-converted video retains only the spikes related to periodic



(a) (b) (c) (d)

(e) (f) (g) (h)

**Fig. 5** Illustrations of the Fourier spectrum in the different analysis stages for *Football* in Fig. 4. The first row is the different analysis stages of Fourier spectrum for the original video: **a** initial, **b** after high-pass filtering, **c** after spike enhancement, and **d** after extracting peak outliers. The second row is the different analysis stages of Fourier spectrum for the up-converted video: **e** initial, **f** after high-pass filtering, **g** after spike enhancement, and **h** after extracting peak outliers

artifacts. As shown in Fig. 5f, once the periodicity is determined, the up-sampling factor $w$ can be derived from the position $f_1$ of the first spike in the normalized frequency domain as follows,

$$w = \frac{1}{f_1}. \tag{11}$$

To avoid that Eq. (10) filters out the first spike, the parameter $d$ should be smaller than $f_1$. Considering that the visual quality of up-converted video, the up-sampling factor $w$ cannot be too large, and the values 2, 4 and 8 are more common. Therefore, the $d$ is set to $\lfloor 0.12 \times N \rfloor$ in Eq. (11), where $\lfloor \cdot \rfloor$ denotes the floor operator.

To highlight the spikes, the spike enhancement is performed to compute the enhanced result $S(k)$ as follows,

$$S(k) = \sum_{l=d+1}^{N-d} \left| F^{\mathrm{H}}(k) - F^{\mathrm{H}}(l) \right|, k = d+1, \cdots, N-d, \tag{12}$$

Figure 5c, g show respectively the enhanced results of original and up-converted videos. It can be seen that the variations of their small spectrum values are smoother than ones of Fig. 5b, f, especially for Fig. 5g, the three spikes look so remarkable in a stationary background, which is more favorable to the automatic extraction of spikes. Finally, we extract the peak outliers from the enhanced result to locate the position of spikes. The flow of extracting peak outliers is summarized in Table 1. At the stage of initialization, the initial peak map $P_0(k)$ is generated by forcing the non-peaks of enhanced result $S(k)$ to be 0. In the main iteration, we regard the peak values lager than 80% of mean value of peaks as the outliers, and neglect those smaller than 80% of mean value of peaks. Until the new outliers cannot occurs, the final peak map $P_{\mathrm{c}}(k)$ can be determined. Figure 5d, h presents respectively $P_{\mathrm{c}}(k)$ of original and up-converted videos. For the original video, the number of outliers is larger, and their magnitudes are smaller. For the up-converted video, the three spikes are only retained at the special positions, and they have the larger magnitudes. Therefore, we can use the number and magnitudes of peak outliers to compute the credibility value as follows,

$$CV = \frac{\mathrm{Max}\{P_{\mathrm{c}}(k) | k = 0, 1, \cdots, N\}}{N_{\mathrm{c}} \times E_0}, \tag{13}$$

**Table 1** The flow of extracting peak outliers

| The Flow of Extracting Peak Outliers |
|---|
| • Initialization:<br>Generate the initial peak map $P_0(k)$ by forcing the non-peaks of enhanced result $S(k)$ to be 0.<br>• Main Iteration: Increment $i$ by 1 and perform the following steps:<br>(a) Compute the mean value $E_i$ of peaks in $P_i(k)$;<br>(b) Do hard-thresholding for $P_i(k)$ as follows,<br>$\tilde{P}_i(k) = \begin{cases} P_i(k), P_i(k) \geq 0.8E_i \\ 0 \quad , P_i(k) < 0.8E_i \end{cases},$<br>(c) If $\{\tilde{P}_i(k) \vert k = 0,1,...,N\} \cap \{\tilde{P}_{i-1}(k) \vert k = 0,1,...,N\} = \varnothing$, stop and set $P_{\mathrm{c}}(k) = \tilde{P}_i(k)$. Otherwise, apply another iteration.<br>• Output: The final peak map $P_{\mathrm{c}}(k)$ including only peak outliers is the $\tilde{P}_i(k)$ after $i$ iterations. |

where Max$\{\cdot\}$ denotes to get the maximum value of data set, $N_c$ is the number of outlier set $\{P_c(k)|k = 0,1,\ldots,N\}$, and $E_0$ is the average of peaks in $P_0(k)$. When the suspected video is the up-converted video, due to the large magnitude and the small number of peak outliers, the *CV* will be larger. However, when the suspected video is the original video, due to the small magnitude and the large number of peak outliers, the *CV* will be smaller. By setting the threshold *T*, if the *CV* is larger than *T*, the video is classified as a tempered one, vice versa. In the proposed detection method, the threshold *T* is only a parameter, and it is important for the accuracy of detection. Based on our experiments, *T* is set to be 1.45. The more details on the setting of *T* will be provided in Section 4.2.

### 3.3 Variations for frame repetition

After some modifications, our method is still applicable to detect the up-converted videos by frame repetition. The frame repetition uses only one adjacent frame to create the inserted frame, which results in that the variances of inserted frames are same with the one of adjacent frame. As shown in Fig. 6a, different from the smooth noise-level curve of original video, the up-converted video by frame repetition has a step-like noise-level curve, of which the jump occurs periodically. Therefore, before the spectral analysis, we firstly compute the gradient field of noise-level curve as follows,

$$g(t) = \sigma(t) - \sigma(t-1), t = 1, 2, \cdots, N, \tag{14}$$

where $\sigma(0) = 0$. Obviously, the gradients in the smooth part of noise-level curve are zero. In order to enhance the periodicity of gradient field, we reset the value 0 in $g(t)$ to be the value 1, i.e.,

$$\overline{g}(t) = \begin{cases} \;\;1 \;\;, g(t) = 0 \\ g(t), g(t) \neq 0 \end{cases}, t = 1, 2, \cdots, N, \tag{15}$$

Figure 6b shows the gradient fields $g(t)$ of both original and up-converted videos. It can be seen that the gradient field of original video still changes smoothly, but there are periodic intense jumps in the gradient field of up-converted video. After analyzing the spectrum of $g(t)$, we get the peak outliers in Fourier domain for original and up-converted videos as shown in Fig. 7a, b. It can be observed that, similar to the results of detecting MCFI, the original video has the more small-magnitude outliers, but the up-converted video retains only the three spikes. Therefore, we can make the detection method described in Section 3.2 be suitable to the frame repetition by adding the following pre-processing,

$$\sigma(t) = \begin{cases} \sigma(t), 1\left\{\overline{g}(t)|t = 1, \cdots, N\right\} < 0.1N \\ \overline{g}(t), 1\left\{\overline{g}(t)|t = 1, \cdots, N\right\} \geq 0.1N \end{cases}, \tag{16}$$

where $\mathbf{1}\{\cdot\}$ denotes to get the number of value 1 in data set. The Eq. (16) means that if the gradient field of noise-level curve contains lots of value 1, we regard that the noise-level curve results from the frame repetition operation.

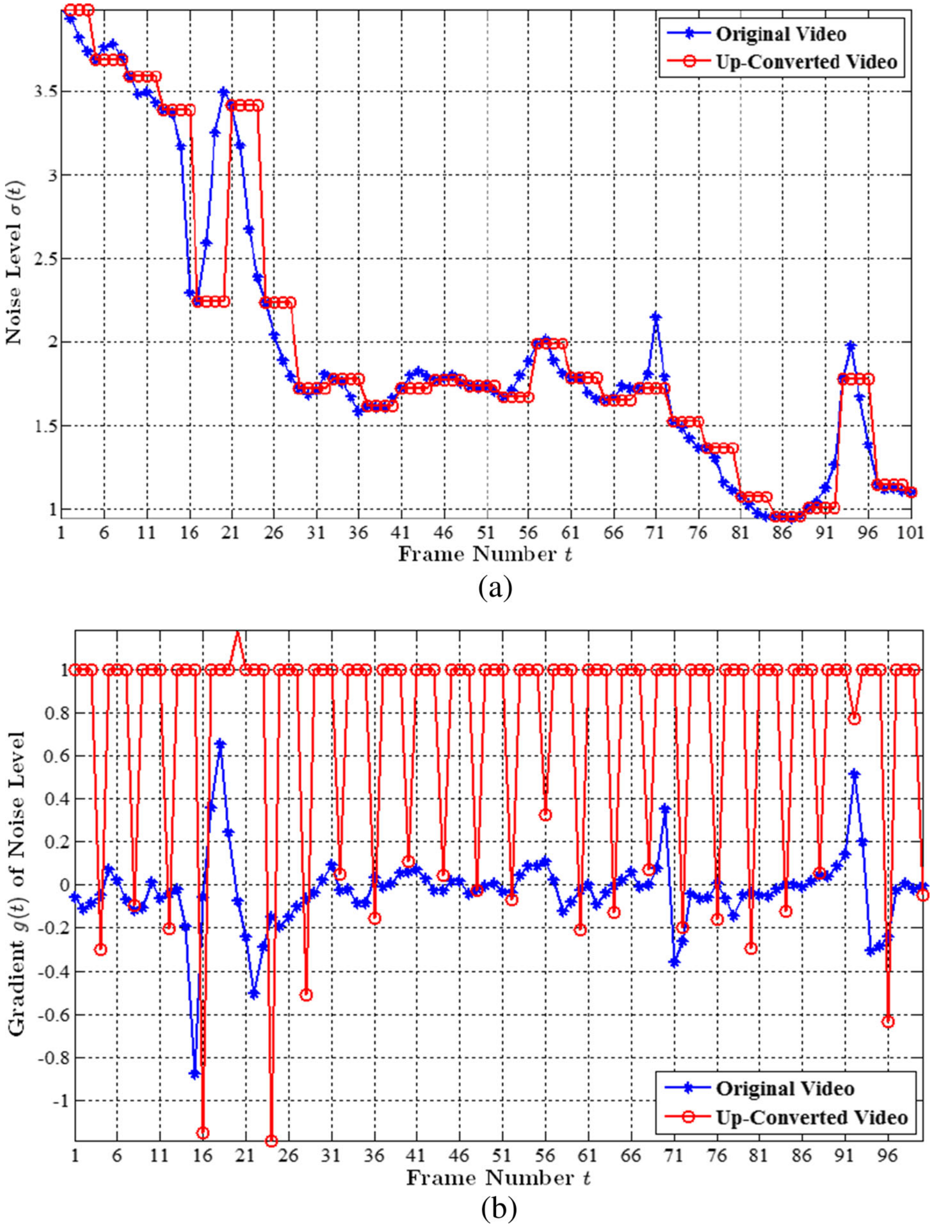


**Fig. 6** Illustrations of noise-level curves and their gradient fields for both original and up-converted *Football* videos. The blue curve represents the original video at 30 fps without up-conversion; and the red curve represents the up-converted video from 7.5 fps to 30 fps by using the frame repetition. **a** Noise-level curves. **b** Gradient curves of noise level

## 3.4 Time-complexity analysis

As described above, the proposed method includes three stages, that is, noise-level estimation of each video frame, spectrum analysis of noise-level curve, and computing

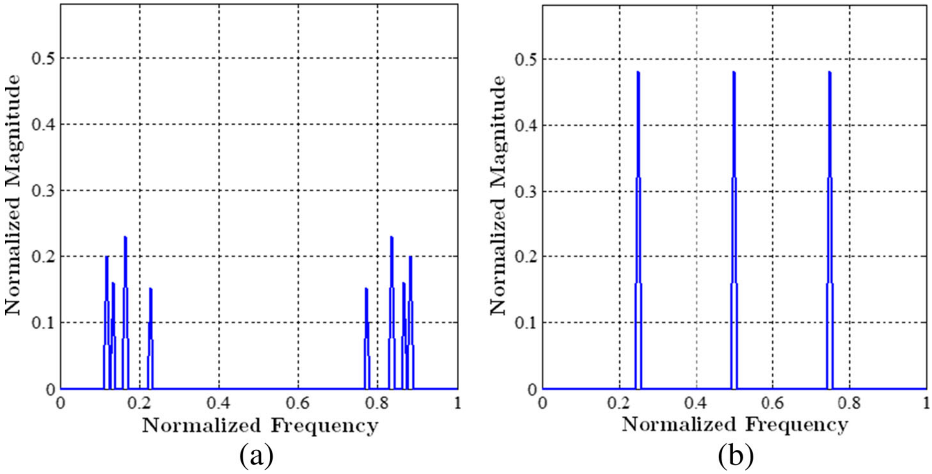


**Fig. 7** Illustrations of peaks outliers in Fourier spectrums of gradient fields. **a** Original video. **b** Up-converted video

the credibility value. We will discuss the time-complexity of the detection method in this section. Assume that the video spatial resolution is $I_r \times I_c$ with $L = I_r \times I_c$ pixels in total, and the frame number of a given video clip is $N$. Table 2 presents the time complexity of each stage in the proposed method. It can be observed from Table 2 that the time complexity of the proposed method mainly depends on the calculation of noise-level estimation in stage one and the FFT in the stage two. In stage one, we use Eq. (8) to compute the noise level of each frame, which costs the time complexity $O(NL\log_2(L))$. The FFT takes the most of computations in the stage two, thus the time complexity of the stage two is $O(N\log_2(N))$. Due to the fact that the number $N_c$ of peak outliers is too small, the total time complexity of the proposed method is $O(NL\log_2(L)) + O(N\log_2(N))$, which means the computation time is related to the video spatial resolution and frame number.

**Table 2** Time complexity analysis of the proposed detection method

| Stage | Algorithm pseudo-code | Time complexity |
|---|---|---|
| 1 | for frame number $t = 1{:}N$ | $N\times[O(L) + O(L\log_2(L))] = O(NL\log_2(L))$ |
| | compute the fine-scale wavelet coefficients $\boldsymbol{y}_s$; | $O(L)$ |
| | compute the noise level $\sigma(t)$ by using Eq. (8); | $O(L\log_2(L))$ |
| | end | |
| 2 | compute the Fourier spectrum $F(k)$ of $\sigma(t)$ using Eq. (10); | $O(N\log_2(N))$ |
| | extract the high-frequency coefficients $F^H(k)$ of $F(k)$ using Eq. (11); | $O(d)$ |
| | compute the enhanced result $S(k)$ of $F^H(k)$ using Eq. (13); | $O(N\text{-}2d)$ |
| | extract peak outliers set $P_c(k)$ by using the flow in Table 1; | $O(N\text{-}2d)$ |
| 3 | compute the credibility value by using Eq. (14); | $O(N_c)$ |

$N$ is the frame number of a given video clip, $L$ is the total pixel number of each video frame, $d$ is the cut-off frequency in high-pass filtering, and $N_c$ is the number of peak outliers in Fourier domain

## 4 Experimental results

In this section, various experiments are conducted to evaluate the performance of the proposed method. Firstly, we construct a training video database to select the best threshold $T$, and then a testing video database is constructed to evaluate the performance of the proposed method. The up-converted video clips are produced by various FRUC algorithms, including the three MCFI methods from [7, 11, 21] and two non-MCFI methods that the frame repetition and the frame averaging. Afterwards, the detection accuracy of our method is compared with that of the MCFI forensics method in [20]. Finally, we present the execution time of the proposed method at the different video spatial resolutions under the following computer configuration:

CPU: Intel(R) Core(TM) i7–3770 @ 3.40 GHz 3.40 GHz
Memory size: 8 GB
OS: Microsoft Windows 7 64 bits
Coding: MATLAB Version 7.6.0.324 (R2008a)

### 4.1 Video database

Two video databases are required to select the important threshold $T$ and evaluate the performance of the proposed method respectively, in which the former is called as the training video database, and the latter is called as the testing video database. As shown in Table 3, the 24 uncompressed YUV sequences[1] with different contents constitutes the basic group of the training video database and the testing video database, and their spatial resolutions are respectively QCIF (176 × 144), CIF (352 × 288), 720P (1280 × 720) and 1080P (1920 × 1080). These original video sequences can further be compressed by H.264/AVC using the following two styles of configuration:

- Configuration 1 (Cfg. 1): the first frame is only the I frame and other frames are the P frame, i.e., it exists no Group of Pictures (GOP), and the Quantization Parameter (QP) is set to be 26, 28 and 30 respectively;
- Configuration 2 (Cfg. 2): insert one I frame every 10 frames, i.e. the length of GOP is 10, and the QP is set to be 26, 28 and 30 respectively.

Due to the I frame having a good visual quality, and the periodicity of I frame has some impacts on the accuracy of our method, which will be discussed in the experiments. The original frame rate of all test video sequences and their compressed versions is 30 fps, and the up-sampling factor $w$ will set to be 2 and 4, e.g., for $w = 2$, the video sequences are firstly down-sampled from 30 fps to 15 fps, and then up-sampled from 15 fps to 30 fps by using the 5 FRUC algorithms including the MCFI methods of [7, 11, 21] and the non-MCFI methods that the frame repetition and the frame averaging. In each video database, the video sequences without FRUC form the Negative Set (NS), and the ones with FRUC form the Positive Set (PS). Above all, the NS of each video database contains the 84 video sequences including 12 original videos and their 72 compressed versions with the Configuration 1 and 2, and the PS of each video database

[1] The uncompressed YUV sequences are coming from the public website: http://media.xiph.org/video/derf/.

**Table 3** The basic group of the training video database and the testing video database

| Spatial Resolution | Training Video database | Testing Video Database |
|---|---|---|
| QCIF (176 × 144) | ice, pairs, harbour | city, crew, soccer |
| CIF (352 × 288) | hall, akiyo, coastguard, stefan, carphone | foreman, football, tennis, mobile, news |
| 720P (1280 × 720) | mobcal, stockholm | ducks_take_off, park_joy |
| 1080P (1920 × 1080) | sunflower, rush_hour | blue_sky, tractor |

contains the 840 video sequences because each negative instance in NS is corresponding to the 10 positive instances with various FRUC methods and up-sampling factors. Combined with the NS and the PS, the two evaluation criterions that False Positive Rate (FPR) and False Negative Rate (FNR) are used, in which the former is the proportion of incorrectly detected ones among all negative instances, and the latter is the proportion of incorrectly detected ones among all positive instances. The average detection accuracy is computed as [100-(*FNR* + *FPR*)/2] %.

## 4.2 Threshold setting

The threshold *T* serves as a criterion of credibility value to determine whether a video has been tampered with MCFI, and it is an important parameter to guarantee the high accuracy of detection. Figure 8 shows the distributions of the credibility values *CV*s of NS and PS in the training video database. It is observed that the *CV*s of all negative instances are smaller than the value of 2, but the nearly 93% values of up-converted videos are distributed over 2. Therefore, we expect that the proper *T* should be less than 2.

Based on the above analysis, we set *T* with different values, where ranges from 0.05 to 2 with a step size of 0.05. To find the best threshold *T*, the NS and PS in the training video database are randomly split into the two non-overlapping subsets equally, i.e., one subset is used for training and another is used for testing. Afterwards, depending on the framework of

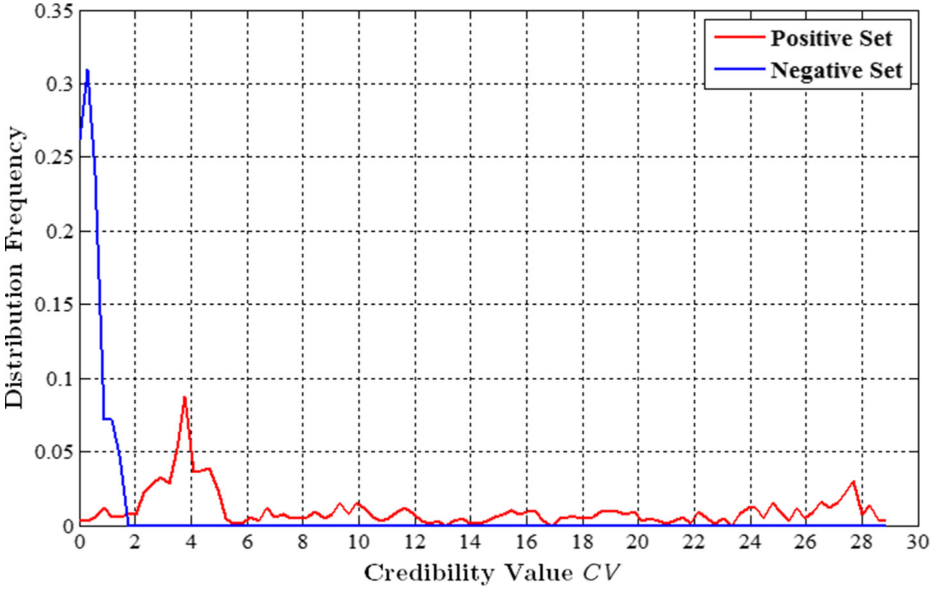
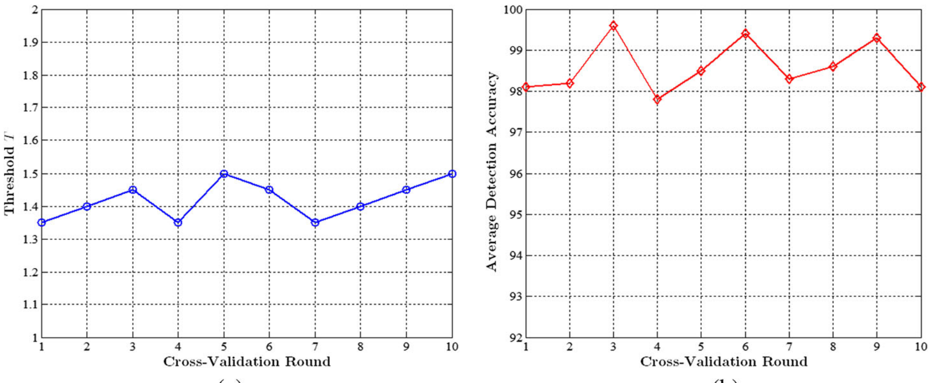


**Fig. 8** The distributions of the credibility values *CV*s of NS and PS in the training video database

cross-validation, we apply the different $T$ values on the training subset to train a reliable threshold under the principle of maximizing the detection accuracy, and evaluate the trained threshold on the testing subset. We repeat the above process ten times and find that the best threshold determined by the training subset is steady for every iteration as shown in Fig. 9a. The ten detection accuracies evaluated on the testing subset are presented in Fig. 9b. It can be seen that the average detection accuracies are all above 97.8% with some deviations, in which the maximum one is 99.6% when the threshold $T$ is 1.45. Based on these experimental results, the threshold $T$ is set to be 1.45 in the proposed method.

## 4.3 Evaluation on MCFI

Table 4 presents the average detection accuracies of the proposed method for various MCFI operators from [7, 11, 21]. It can be seen that the average detection accuracies for these up-converted videos by [7] and [21] are 100%, which proves that our method can successfully identify the up-converted videos by those MCFI methods equipped with the linear MCI operator (i.e., Eq. (1)). However, [11] performs the non-linear MCI to jointly generate the interpolated frames based on the motion vector fields with difference densities, and then uses the texture optimization to further improve the interpolated results, which interferes with the noise accumulation in Eq. (1). Therefore, we can observe from Table 4 that the average detection accuracies on total video sequences are 98.22% and 88.89% for up-sampling factor $w = 2$ and 4 respectively, especially for these up-converted videos compressed with Cfg. 1, the average detection accuracy is only 86.11% when the up-sampling factor $w$ is set to be 4, which proves that our method can lose some precision for those MCFI methods equipped with the non-linear MCI operator and some post-processing. Though some failures occur when detecting the MCFI of [11], it can be seen from the last row of Table 4 that the average detection accuracy of the total testing database on all MCFI methods is still up to 99.41% and 96.23% for $w = 2$ and 4 respectively, which verifies the validity of the proposed method for the MCFI operator. Figure 10 shows the analysis results of the proposed method for original and up-converted videos with compression parameters of QP = 30 and GOP = 10, and the test video sequences are respectively are *crew*, *news*, *park_joy* and *tractor*. Firstly, we can see from Fig. 10a that it exists no salient peaks in the Fourier domain of those videos without MCFI, but



(a) (b)

**Fig. 9** Results of cross validation. **a** The best thresholds for ten iterations of training. **b** Detection accuracies for ten iterations of testing

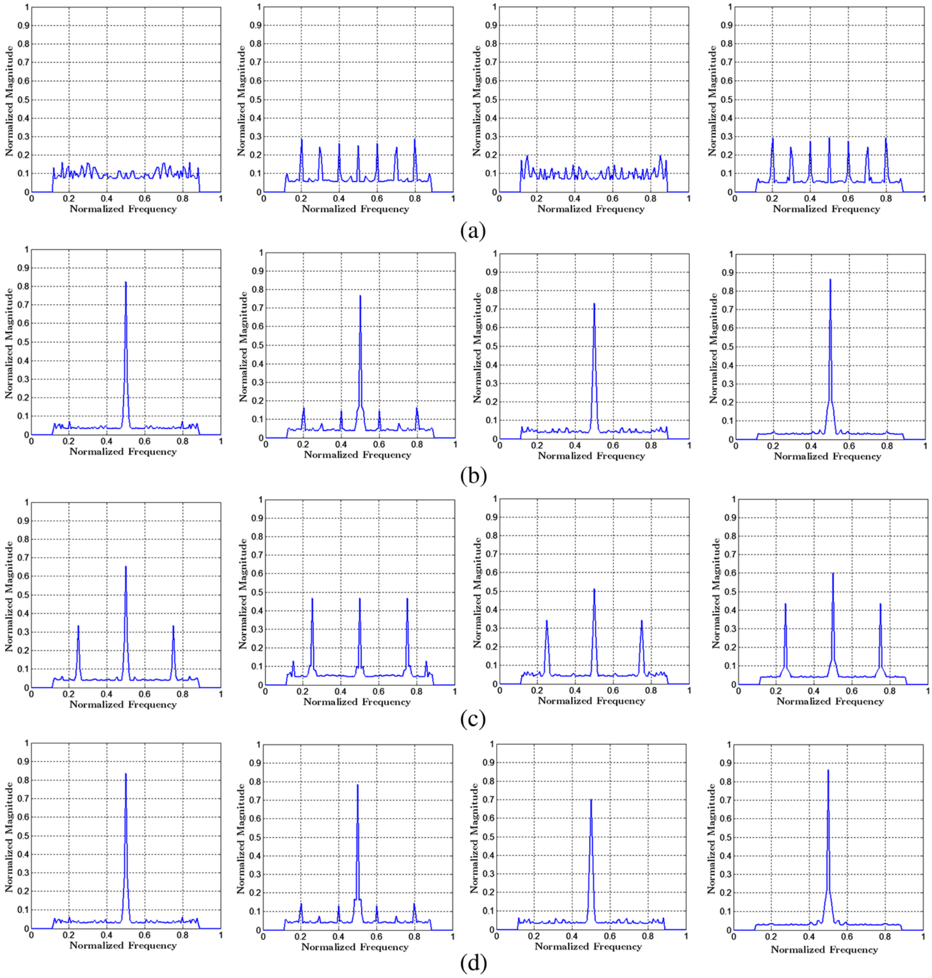
**Table 4** Average detection accuracies (%) of the proposed detection method for various MCFI operators

| MCFI Method | Style of Test Video | FPR (%) | FNR (%) | | Detection Accuracy (%) | |
|---|---|---|---|---|---|---|
| | | | 15 fps to 30 fps ($w = 2$) | 7.5 fps to 30 fps ($w = 4$) | 15 fps to 30 fps ($w = 2$) | 7.5 fps to 30 fps ($w = 4$) |
| Dikbas et al. [7] | Uncompressed | 0 | 0 | 0 | 100 | 100 |
| | Compressed with Cfg. 1 | 0 | 0 | 0 | 100 | 100 |
| | Compressed with Cfg. 2 | 0 | 0 | 0 | 100 | 100 |
| | Total | 0 | 0 | 0 | 100 | 100 |
| Yoo et al. [21] | Uncompressed | 0 | 0 | 0 | 100 | 100 |
| | Compressed with Cfg. 1 | 0 | 0 | 0 | 100 | 100 |
| | Compressed with Cfg. 2 | 0 | 0 | 0 | 100 | 100 |
| | Total | 0 | 0 | 0 | 100 | 100 |
| Jeong et al. [11] | Uncompressed | 0 | 0 | 8.33 | 100 | 95.84 |
| | Compressed with Cfg. 1 | 0 | 0 | 27.78 | 100 | 86.11 |
| | Compressed with Cfg. 2 | 0 | 8.33 | 22.22 | 95.84 | 88.89 |
| | Total | 0 | 3.57 | 22.62 | 98.22 | 88.69 |
| Total | Uncompressed | 0 | 0 | 2.28 | 100 | 98.86 |
| | Compressed with Cfg. 1 | 0 | 0 | 9.26 | 100 | 95.37 |
| | Compressed with Cfg. 2 | 0 | 2.78 | 7.41 | 98.61 | 96.30 |
| | Total | 0 | 1.19 | 7.54 | 99.41 | 96.23 |

some moderate-magnitude peaks occur regularly for the *news* and *tractor* sequences with slow motions, which results from the periodic insertion of I frames. By Fig. 10a–e, it is observed that the analysis results have the salient spikes for the forged videos with MCFI of [7, 21], and they cannot be impacted by the insertion of I frames. However, for the up-converted videos by [11], we can see from Fig. 10f, g that the magnitudes of spikes for several test video sequences are cut down, which results in a low credibility value as so to cause the mistakes. Though some forged videos with MCFI of [11] have the weaker periodicity of noise-level curve, we still observe visually the peak outliers in Fig. 10f, g so as to make a correct judgment.

## 4.4 Evaluation on non-MCFI

Our detection can be still used to identify the up-converted videos by the non-MCFI methods including the frame repetition and the frame averaging, and the average detection accuracies are presented in the Table 5. It can be seen that the average detection accuracies for these up-converted videos by the frame repetition are 100%, which proves that our method can successfully identify the frame repetition operator. Figure 11a, b show also the analysis results for the frame repetition, and we can see that it exists the salient spikes. However, for the frame averaging, some failures occur in the PS of test video database, especially for the conditions of compression configuration 2 and $w = 4$, the average detection accuracy decreases to 80.55%. Figure 11c shows the analysis results for frame averaging when $w = 2$, and we can observe that the spikes are still obvious. However, as shown in Fig. 11d, the interference with the magnitude and the position of spikes is strong in the analysis results for frame averaging when $w = 4$. The bad performance for detecting the frame averaging results from the non-stationary local statistical characteristics of pixel values along the time axis, i.e., the pixel values at the same position on the time axis change quickly, therefore the statistical distributions of these pixels have the larger differences, which will lead to the failure of Eq. (7). Due to the fact that the statistical characteristics of the temporal neighboring pixels has locally



(a)

(b)

(c)

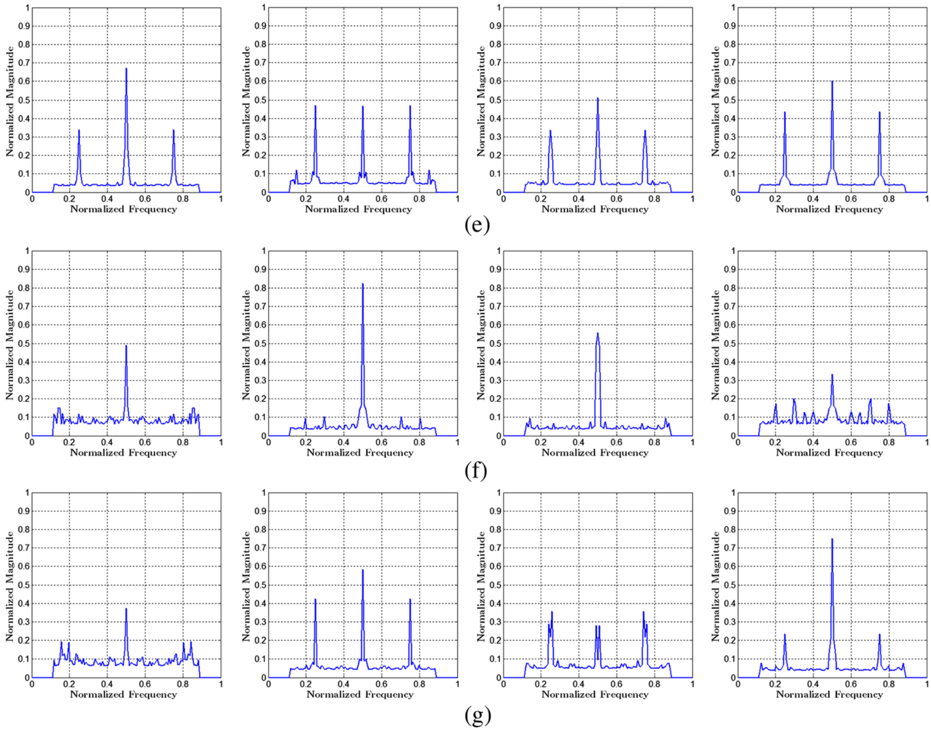
(d)

**Fig. 10** Analysis results of the proposed method for original and up-converted videos with compression parameters of QP = 30 and GOP = 10: **a** original, **b** up-converted by [7] with $w = 2$, **c** up-converted by [7] with $w = 4$, **d** up-converted by [21] with $w = 2$, **e** up-converted by [21] with $w = 4$, **f** up-converted by [11] with $w = 2$, and **g** up-converted by [11] with $w = 4$. In each subfigure, the test videos are *crew*, *news*, *park_joy* and *tractor* from left to right

stationary along with the motion trajectory, the interpolated frame by the linear MCI has a significant decrease of noise level when compared with ones of adjacent frames. Considering that the frame averaging has a poor interpolated quality than those of [7, 11, 21] as shown in Fig. 12, the forger cannot often use it to up-convert videos, therefore we do not require to be particularly concerned with the low detection accuracy for frame averaging.

## 4.5 Performance comparison

We compare the proposed detection method with the work in [20], and Table 6 summaries their average detection accuracies for various MCFI and non-MCFI operators. It can be seen that when detecting the MCFI methods, the method of Yao et al. [20] has the poorer



(e)

(f)

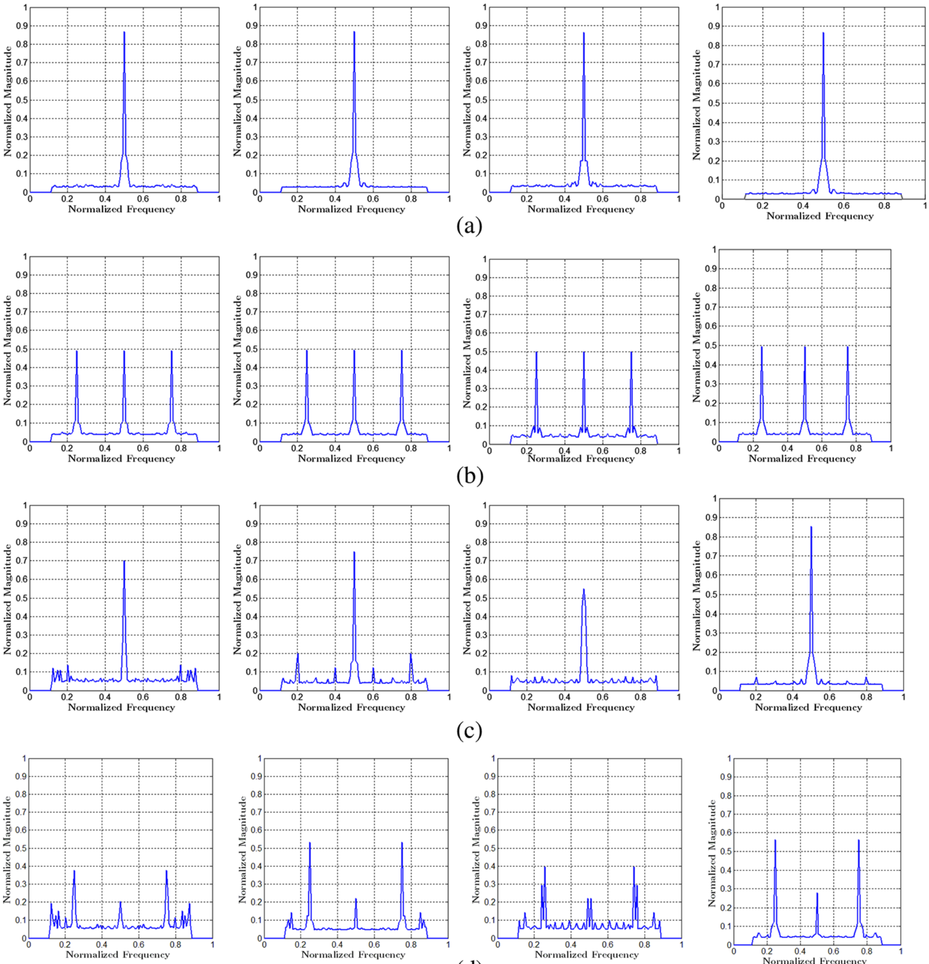
(g)

**Fig. 10** (continued)

performance compared with the results of our detection method, e.g., its average detection accuracy of detecting the MCFI methods [7, 21] with the linear MCI operator can only be up to be 87.50%. However, our method perfectly identifies authenticities of all test video. For the MCFI method [11] with non-linear MCI operator, the average detection accuracies of [20] on total video sequences are 70.84% and 83.93% when the up-sampling factor $w$ is set to be 2 and 4 respectively, and these results still have some gaps compared with the detection accuracies of our method. As shown in Table 7, when detecting the original video sequences, the failure rate

**Table 5** Average detection accuracies (%) for the frame repetition and the frame averaging

| Non-MCFI Method | Style of Test Video | FPR (%) | FNR (%) | | Detection Accuracy (%) | |
|---|---|---|---|---|---|---|
| | | | 15 fps to 30 fps ($w$ = 2) | 7.5 fps to 30 fps ($w$ = 4) | 15 fps to 30 fps ($w$ = 2) | 7.5 fps to 30 fps ($w$ = 4) |
| Frame Repetition | Uncompressed | 0 | 0 | 0 | 100 | 100 |
| | Compressed with Cfg. 1 | 0 | 0 | 0 | 100 | 100 |
| | Compressed with Cfg. 2 | 0 | 0 | 0 | 100 | 100 |
| | Total | 0 | 0 | 0 | 100 | 100 |
| Frame Averaging | Uncompressed | 0 | 0 | 0 | 100 | 100 |
| | Compressed with Cfg. 1 | 0 | 2.78 | 30.56 | 98.61 | 84.72 |
| | Compressed with Cfg. 2 | 0 | 2.78 | 38.89 | 98.61 | 80.55 |
| | Total | 0 | 2.38 | 29.76 | 98.81 | 85.12 |



(a)

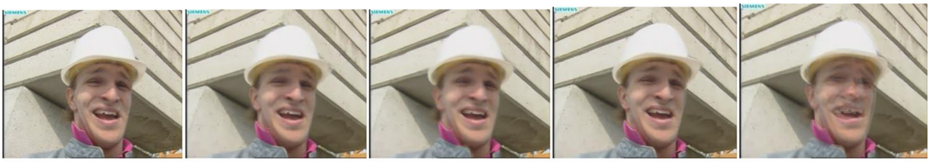
(b)

(c)

(d)

**Fig. 11** Analysis results of the proposed method for the up-converted videos with compression parameters of QP = 30 and GOP = 10: **a** up-converted by the frame repetition with $w$ = 2, **b** up-converted by the frame repetition with $w$ = 4, **c** up-converted by the frame averaging with $w$ = 2, **d** up-converted by the frame averaging with $w$ = 4. In each subfigure, the test videos are *crew*, *news*, *park_joy* and *tractor* from left to right

of method in [20] is 25%, (i.e., FPR = 25%), and lots of failures also occurs when detecting the up-converted video sequences with MCIF methods. The high failure rates in NS and PS both



(a) (b) (c) (d) (e)

**Fig. 12** Visual results on the 15th interpolated frame of *Foreman* sequence with different FRUC algorithms: **a** the ground truth of the interpolated frame, **b** interpolated by [7], **c** interpolated by [21], **d** interpolated by [11], and **e** interpolated by the frame averaging

**Table 6** Average detection accuracies (%) of the existing detection methods for various MCFI and non-MCFI methods

| Forgery Method | Style of Test Video | Detection Accuracy (%) | | | |
|---|---|---|---|---|---|
| | | 15 fps to 30 fps ($w$ = 2) | | 7.5 fps to 30 fps ($w$ = 4) | |
| | | Yao et al. [20] | Proposed | Yao et al. [20] | Proposed |
| Dikbas et al. [7] | Uncompressed | 75.00 | 100 | 87.50 | 100 |
| | Compressed with Cfg. 1 | 73.61 | 100 | 83.34 | 100 |
| | Compressed with Cfg. 2 | 75.00 | 100 | 83.34 | 100 |
| | Total | 74.41 | 100 | 83.93 | 100 |
| Yoo et al. [21] | Uncompressed | 83.34 | 100 | 87.50 | 100 |
| | Compressed with Cfg. 1 | 81.95 | 100 | 87.50 | 100 |
| | Compressed with Cfg. 2 | 81.95 | 100 | 87.50 | 100 |
| | Total | 82.13 | 100 | 87.50 | 100 |
| Jeong et al. [11] | Uncompressed | 70.84 | 100 | 83.34 | 95.84 |
| | Compressed with Cfg. 1 | 70.84 | 100 | 83.34 | 86.11 |
| | Compressed with Cfg. 2 | 70.84 | 95.84 | 84.72 | 88.89 |
| | Total | 70.84 | 98.22 | 83.93 | 88.69 |
| Frame Repetition | Uncompressed | 45.84 | 100 | 37.50 | 100 |
| | Compressed with Cfg. 1 | 41.67 | 100 | 37.50 | 100 |
| | Compressed with Cfg. 2 | 44.44 | 100 | 37.50 | 100 |
| | Total | 43.45 | 100 | 37.50 | 100 |
| Frame Averaging | Uncompressed | 87.50 | 100 | 87.50 | 100 |
| | Compressed with Cfg. 1 | 87.50 | 98.61 | 87.50 | 84.72 |
| | Compressed with Cfg. 2 | 87.50 | 98.61 | 87.50 | 80.55 |
| | Total | 87.50 | 98.81 | 87.50 | 85.12 |

reduce the detection accuracy of [20], and these larger accuracy losses proves that the Kaufman adaptive moving average proposed by [20] cannot still extract a robust adaptive threshold from the edge-intensity curve of suspected video sequence, and the edge losses in the interpolated frames might not be an effective clue to detect MCFI operator. For non-MCFI methods, it can be seen from Table 6 that the method of [20] has a low detection accuracies on all test videos for the frame repetition, and Table 7 also presents the FNR values of [20] are larger than 83.33% when detecting the up-converted videos with frame repetition, which indicates that the method of [20] is suitable to detecting the frame repetition. However, the method of [20] has the better performance when detecting the up-converted videos with frame averaging, e.g., when the up-sampling factor $w$ is set to be 4, the method in [20] outperforms our method, and its average detection accuracy on total test videos can be up to 87.50%. Table 7 presents FNR values of [20] are 0% for any style of test video, i.e., the method of [20] can perfectly identify the authenticities of the up-converted videos with frame averaging.

### 4.6 Execution time results

Table 8 shows the average execution times of the proposed method and [20] for the test videos at the different spatial resolutions. The length of all test videos is 100. We can see that the average execution time of two detection methods will increase as the spatial resolution increases, e.g., when detecting a video sequence with QCIF format, our method and [20] requires 0.44 s and 0.59 s respectively, and when detecting a video sequence with 1080P format, our method and [20] requires 46.63 s and 32.60 s respectively. From the last row of Table 8, it can be seen that the average runtime of each frame on all spatial resolutions is only

**Table 7** FPR and FNR values (%) of the existing detection methods for various MCFI and non-MCFI methods

| Forgery Method | Style of Test Video | FPR (%) | | FNR (%) | | | |
|---|---|---|---|---|---|---|---|
| | | | | 15 fps to 30 fps ($w$ = 2) | | 7.5 fps to 30 fps ($w$ = 4) | |
| | | Yao et al. [20] | Proposed | Yao et al. [20] | Proposed | Yao et al. [20] | Proposed |
| Dikbas et al. [7] | Uncompressed | 25.00 | 0 | 25.00 | 0 | 0 | 0 |
| | Compressed with Cfg. 1 | 25.00 | 0 | 27.78 | 0 | 8.33 | 0 |
| | Compressed with Cfg. 2 | 25.00 | 0 | 25.00 | 0 | 8.33 | 0 |
| | Total | 25.00 | 0 | 26.19 | 0 | 7.14 | 0 |
| Yoo et al. [21] | Uncompressed | 25.00 | 0 | 8.33 | 0 | 0 | 0 |
| | Compressed with Cfg. 1 | 25.00 | 0 | 11.11 | 0 | 0 | 0 |
| | Compressed with Cfg. 2 | 25.00 | 0 | 11.11 | 0 | 0 | 0 |
| | Total | 25.00 | 0 | 10.74 | 0 | 0 | 0 |
| Jeong et al. [11] | Uncompressed | 25.00 | 0 | 33.33 | 0 | 8.33 | 8.33 |
| | Compressed with Cfg. 1 | 25.00 | 0 | 33.33 | 0 | 8.33 | 27.78 |
| | Compressed with Cfg. 2 | 25.00 | 0 | 33.33 | 8.33 | 5.56 | 22.22 |
| | Total | 25.00 | 0 | 33.33 | 3.57 | 7.14 | 22.62 |
| Frame Repetition | Uncompressed | 25.00 | 0 | 83.33 | 0 | 100 | 0 |
| | Compressed with Cfg. 1 | 25.00 | 0 | 91.67 | 0 | 100 | 0 |
| | Compressed with Cfg. 2 | 25.00 | 0 | 86.11 | 0 | 100 | 0 |
| | Total | 25.00 | 0 | 88.11 | 0 | 100 | 0 |
| Frame Averaging | Uncompressed | 25.00 | 0 | 0 | 0 | 0 | 0 |
| | Compressed with Cfg. 1 | 25.00 | 0 | 0 | 2.78 | 0 | 30.56 |
| | Compressed with Cfg. 2 | 25.00 | 0 | 0 | 2.78 | 0 | 38.89 |
| | Total | 25.00 | 0 | 0 | 2.38 | 0 | 29.76 |

0.170 s for our method, and the method of [20] requires 0.124 s to detect a frame, which proves that the our method cannot increase excessive calculations while improving the detection accuracy when compared with the method of [20].

From Table 8, we can observe that the detection speed of our method is not satisfactory, especially for the high-definition video. Therefore, we expect to reduce the computational complexity while guaranteeing a high detection accuracy. A simple approach is to perform our detection method on small spatial windows for each frame, i.e., cropping the original frames and using these smaller frames to estimate noise-level curve. For this purpose, we evaluate our method on the CIF video sequences using a square spatial window ranging from 10 × 10 to 200 × 200

**Table 8** Average execution times of the existing detection methods for the test videos at the different spatial resolutions

| Spatial Resolution | Execution Time | | | |
|---|---|---|---|---|
| | Total Time (s) | | Time (s/frame) | |
| | Yao et al. [20] | Proposed | Yao et al. [20] | Proposed |
| QCIF (176 × 144) | 0.59 | 0.44 | 0.005 | 0.004 |
| CIF (352 × 288) | 1.49 | 1.69 | 0.015 | 0.016 |
| 720P (1280 × 720) | 15.11 | 20.16 | 0.151 | 0.199 |
| 1080P (1920 × 1080) | 32.60 | 46.63 | 0.326 | 0.462 |
| Avg. | 12.45 | 17.23 | 0.124 | 0.170 |

pixels. Figure 13 shows the detection accuracy and execution time averaged on the all testing video sequences when the different up-sampling factors are used. We can see that the average detection accuracy gradually increases as the window size increases, and there is a low exaction time when using a smaller window. Indeed, a big window contains the more pixel samples, thus the estimator of noise level is closer to the truth. On the other hand, using a smaller window, the number of pixel samples is limited, and the analysis on noise-level curve leads to some incorrect results. However, a moderate window size can both guarantee a high detection accuracy and a low computational complexity, e.g., when window size is 100, the average detection accuracy is more than 90%, and it only requires 0.2 s to detect a CIF video sequence. Besides, we notice that for the different up-sampling factors, our method seems to be so robust to spatial cropping.

## 5 Conclusions

In this paper, we propose an effective method to expose the fake high-frame-rate videos forged by MCFI. Based on the discovery that the MCFI interpolates the new frames by averaging the pixel values of adjacent frames, the specific noise-level variation in the interpolated frame is used to expose the possible MCFI operator. These inserted frames are presented periodically, and therefore our detection method is to determine whether there exists periodicity or not for those smaller noise levels in a suspected video clip. To automatically identify the up-converted videos, some spectrum analysis tools are performed to extract the salient spikes in the Fourier domain, and then we use these salient spikes to design the criterion of credibility value. Finally, depending on this criterion, a robust hard-thresholding is used to make a decision. The experimental results evaluated on the test video sequences at different spatial resolutions have shown the effectiveness of the proposed method. The average detection accuracy can be up to 100% for these up-converted videos by the MCFI method and frame repetition in uncompressed and H.264/AVC format. Besides, the proposed method has a low computational complexity, and its average execution time of each frame is only 0.170 s for some common spatial resolutions.

The proposed method determines the threshold parameter through the cross-validation in training video database, and however the limitation of the training set will also suppress the widely application of our method. Therefore, an adaptive threshold setting is required to be

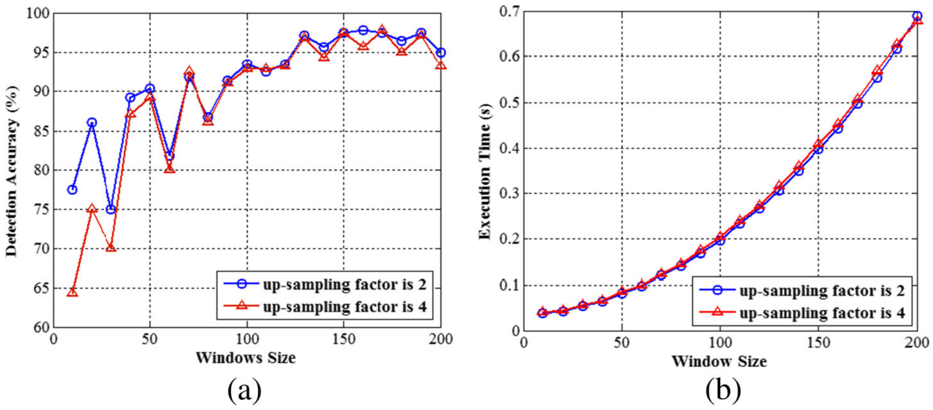


**Fig. 13** Average detection accuracy and execution time for different window size (in pixels) when using the different up-sampling factors: **a** average detection accuracy, and **b** average execution time

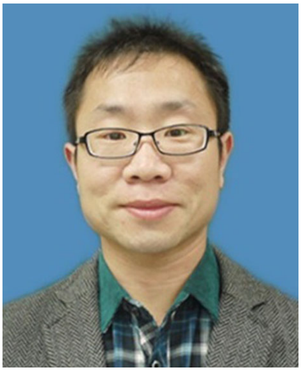
studied in the future, and a possible solution may be some ideas similar to the spectral segmentation used in the spectral clustering.

**Acknowledgements** This work was supported in part by the National Natural Science Foundation of China, under Grants nos. 61501393, 61572417, 61502409 and 61471162.

## References

1. Alsmirat M, Jararweh Y, Al-Ayyoub M, Gupta BB (2016a) Accelerating compute intensive medical imaging segmentation algorithms using hybrid CPU-GPU implementations. Multimedia Tools and Applications 2016:1–19
2. Alsmirat MA, Jararweh Y, Obaidat I, Gupta BB (2016b) Automated wireless video surveillance: an evaluation framework. J Real-Time Image Proc 2016:1–20
3. Bestagini P, Battaglia S, Milani S, Tagliasacchi M, Tubaro S (2013) Detection of temporal interpolation in video sequences. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp 3033–3037
4. Bian S, Luo W, Huang J (2014a) Detecting video frame-rate up-conversion based on periodic properties of inter-frame similarity. Multimedia Tools and Applications 72(1):437–451
5. Bian S, Luo W, Huang J (2014b) Exposing fake bit rate videos and estimating original bit rates. IEEE transactions on circuits and Systems for Video. Technology 24(12):2144–2154
6. Choi BD, Han JW, Kim CS, Ko SJ (2007) Motion-compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation. IEEE transactions on circuits and Systems for Video. Technology 17(4):407–416
7. Dikbas S, Altunbasak T (2013) Novel true-motion estimation algorithm and its application to motion-compensated temporal frame interpolation. IEEE Trans Image Process 22(8):2931–2945
8. Donoho DL (1995) De-noising by soft-thresholding. IEEE Trans Inf Theory 41(3):613–627
9. Fu Z, Wu X, Guan C, Sun X, Ren K (2016) Towards efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement. IEEE Trans Inf Forensics Secur. doi:10.1109/TIFS.2016.2596138
10. Haan GD, Biezen PWAC, Huijgen H, Ojo OA (1993) True motion estimation with 3-D recursive search block matching. IEEE transactions on circuits and Systems for Video. Technology 3(5):368–379
11. Jeong SG, Lee C, Kim CS (2013) Motion-compensated frame interpolation based on multihypothesis motion estimation and texture optimization. IEEE Trans Image Process 22(11):4497–4509
12. Li J, Li X, Yang B, Sun X (2015) Segmentation-based image copy-move forgery detection scheme. IEEE Trans Inf Forensics Secur 10(3):507–518
13. Liu HB, Xin RQ, Zhao DB, Ma SW, Gao W (2012) Multiple hypotheses bayesian frame rate up-conversion by adaptive fusion of motion-compensated interpolations. IEEE transactions on circuits and Systems for Video. Technology 22(8):1188–1198
14. Mallat S (1989) A theory for multiresolution signal decomposition: the wavelet representation. IEEE Trans Pattern Anal Mach Intell 11(7):674–693
15. Mehmood I, Sajjad M, Rho S, Baik SW (2015) Divide-and-conquer based summarization framework for extracting affective video content. Nerocomputing 174:393–403
16. Pan Z, Zhang Y, Kwong S (2015) Efficient motion and disparity estimation optimization for low complexity multiview video coding. IEEE Trans Broadcast 61(2):166–176
17. Pan Z, Lei J, Zhang Y, Sun X, Kwong S (2016) Fast motion estimation based on content property for low-complexity H.265/HEVC encoder. IEEE Trans Broadcast 62(3):675–684
18. Papoulis A, Pillai SU (2002) Probability, random variables and stochastic processes, 4th ed. McGraw-Hill, New York
19. Xia Z, Wang X, Zhang L, Qin Z, Sun X, Ren K (2016) A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. IEEE Trans Inf Forensics Secur. doi:10.1109/TIFS.2016.2590944
20. Yao Y, Yang G, Sun X, Li L (2016) Detecting video frame-rate up-conversion based on periodic properties of edge-intensity. Journal of Information Security and Applications 26:39–50
21. Yoo DG, Kang SJ, Kim YH (2013) Direction-select motion estimation for motion-compensated frame rate up-conversion. J Disp Technol 9(10):840–850
22. Zhou Z, Wang Y, Wu QMJ, Yang C, Sun X (2016) Effective and efficient global context verification for image copy detection. IEEE Trans Inf Forensics Secur. doi:10.1109/TIFS.2016.2601065

**Ran Li** was born in 1988. He received Ph.D. degrees from the School of Communication and Information Engeerning, Nanjing University of Posts and Telecommunications, Nanjing, China, in 2014. He currently works as Lecturer with the School of Computer and Information Technology, Xinyang Normal University, Xinyang, China. His current research interests include multimedia information security and video forensics.



**Zhenghui Liu** was born in 1983. He received Ph.D. degrees from the School of Information Science and Technology, Southwest Jiaotong University, Chengdu, in 2014. He currently works as Lecturer with the School of Computer and Information Technology, Xinyang Normal University, Xinyang, China. His current research interests include multimedia information security and audio forensics.

**Yu Zhang** was born in 1986. He received Ph.D. degrees from the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China, in 2014. He currently works as Lecturer with the School of Computer and Information Technology, Xinyang Normal University, Xinyang, China. His current research interests include information security and public key encryption.

**Yanling Li** was born in 1975. He received Ph.D. degrees from the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China, in 2009. She currently works as Lecturer with the School of Computer and Information Technology, Xinyang Normal University, Xinyang, China. Her current research interests include image processing and pattern recognition.

**Zhangjie Fu** was born in 1983. He received Ph.D. degrees from the School of Information Science and Technology, Hunan University, Changsha, China, in 2012. He currently works as Associate Profession with the School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing, China. Her current research interests include cloud computing, and dig data security.