Research Article

Robust Frame Duplication Detection for Degraded Videos

Qi Han ¹, Hao Chen,¹ Liyang Yu,² and Qiong Li¹

¹School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

²School of Software Engineering, Harbin University and Science and Technology, Harbin 150080, China

Correspondence should be addressed to Qi Han; qi.han@hit.edu.cn

Received 31 December 2020; Revised 16 March 2021; Accepted 8 April 2021; Published 21 April 2021

Academic Editor: Liguozhang

Copyright © 2021 Qi Han et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

To detect frame duplication in degraded videos, we proposed a coarse-to-fine approach based on locality-sensitive hashing and image registration. The proposed method consists of a coarse matching stage and a duplication verification step. In the coarse matching stage, visually similar frame sequences are preclustered by locality-sensitive hashing and considered as potential duplication candidates. These candidates are further checked by a duplication verification step. Being different from the existing methods, our duplication verification does not rely on a fixed distance (or correlation) threshold to judge whether two frames are identical. We resorted to image registration, which is intrinsically a global optimal matching process, to determine whether two frames coincide with each other. We integrated the stability information into the registration objective function to make the registration process more robust for degraded videos. To test the performance of the proposed method, we created a dataset, which consists of 3 subsets of different kinds of degradation and 117 forged videos in total. The experimental results show that our method outperforms state-of-the-art methods for most cases in our dataset and exhibits outstanding robustness under different conditions. Thanks to the coarse-to-fine strategy, the running time of the proposed method is also quite competitive.

1. Introduction

With various nonlinear editing tools such as Adobe Premiere, Microsoft Movie Maker, and Sony Vegas, it is now much easier for people to tamper the content of a video. Many different kinds of detection methods have been proposed [1–3]. Among different approaches to video forgery, frame duplication, which simply copies a sequence of frames to another position in the timeline, may be one of the most convenient yet effective means to hide or counterfeit events. The forged part of the video can be easily made visually natural and therefore difficult for manual detection. Fortunately, since the source and target frames simultaneously exist in the video, frame duplication forgery can be exposed by detecting abnormal identical frame sequences. On this basis, several methods have been proposed [4–8]. These methods share a common methodology to judge whether a frame is the copy of another one. These methods share a common methodology to judge whether a frame is the copy of another one. They extract features from the frames and set the distance threshold between the features. Such methodology makes it difficult for

these methods to perform robustly when applied in realistic frame duplication detection (FDD), where degradation is quite common. In degraded videos, the local structure of the frames can be altered slightly by sorts of factors, and then the value of the extracted features correspondingly changes; therefore, a fixed distance threshold cannot always work well. For instance, an experienced attacker may add a little perturbation (e.g., additive noise) to either the source or the target frames; then, in this noisy scenario, it is quite probable that the threshold tuned for ordinary cases will miss some true matching frame pairs. In fact, even the lossy encoding process will result in substantial difference between the source and target frames; please see Figure 1 for an example. Since a tampered video will be subject to at least double compressions, this example indicates that degradation in video forgeries is almost unavoidable. This makes it quite complicated to stably detect frame duplication in realistic scenarios.

In this paper, we are attempting quite a different methodology for FDD. In the proposed method, we no longer rely on a fixed threshold to decide whether a video subsequence is the duplication of another. The key idea is

FIGURE 1: An example demonstrating the substantial difference between a frame and its copy. Such difference is caused only by the lossy encoding process itself. The video is H.264 encoded. (a, b) A pair of source and target frames. Although they are visually identical, nonnegligible error between them has been introduced during the encoding process. (c) The histogram of pixel-wise absolute difference. We can see that the error can even be as large as 30, and the number of occurrences of the errors which are larger than or equal to 10 is over 700,000 (the resolution of the video is 1920×1080). To further demonstrate the impact of such difference, we build a 500-word vocabulary by k -means clustering 7500 dense SIFT features extracted from 72 randomly selected images, and the vocabulary is lexicographically sorted so that neighbouring visual words in the vocabulary are also nearer in the feature space. We then extract dense SIFT features for each pixel in (a) and (b) and map the features to the indices of the visual words so that we get two visual word index maps (d, e) and the absolute difference map between them (f). The considerable amount of bright spots in (f) indicates that the lossy compression substantially changed the local structure and hence local feature.

that for any two frames f_a and f_b , if they contain the same objects, meanwhile the corresponding objects' shape and position are identical, f_a and f_b can be considered as the copy of each other. We resort to image registration to check whether the three aspects (objects, objects' shape, and objects' position) of two frames coincide with their counterparts. Specifically, the problem is solved by pixel-level global optimal matching. When a given frame is aligned to its copy,

there should not be any distortion in the resulted offset field, i.e., each pixel in the source frame matches to the same location in the target frame. Our method is robust to certain magnitude of video degradation, but the global matching procedure is not so fast as some feature extraction and comparison-based methods such as [4]. For acceleration, our pipeline involves a coarse matching step, which can significantly improve the computational speed.

It should be noted that, like the other methods in this field [4–8], our method also does not take the static scenes into consideration. Therefore, our method can be used to expose counterfeit events. For instance, an attacker can copy a sequence of frames from the historical record to counterfeit the event that a man passing through a scene; then, the duplicated frames can be used as the evidence of being present or absent and therefore plant suspicion on that man or absolve him from guilty.

The contributions of this paper are listed as follows.

First, we propose a coarse-to-fine FDD scheme, whose first key step is preclustering perceptually similar sequences of frames by locality-sensitive hashing (LSH). Through this coarse matching step, the computation load for finer duplication verification can be reduced by several orders of magnitude.

Second, we use global optimal matching for finer duplication verification, with the benefit regarding computational cost obtained in the coarse matching step. We integrate the stability information of different regions into the matching objective, resulting in a robust yet sensitive matcher for the noisy environment.

The rest of this paper is organized as follows: in section “Related Work,” we briefly introduce related work, and then in section “Proposed Method,” the proposed method is detailed. The experimental results are presented in section “Experimental Results.” The conclusion and future directions are finally drawn in section “Conclusion and Future work.”

2. Related Work

To the best of our knowledge, there are only a few works on FDD. In [8], Wang and Farid proposed the first FDD method. The video is divided into overlapping short subsequences; then, for each subsequence, temporal correlation between each pair of frames and the block-wise spatial correlation within each frame are calculated and used as features for subsequence comparison. The subsequences are compared with each other in a coarse-to-fine manner. High similarity in temporal correlation coefficients will trigger the comparison between spatial correlation coefficients [9]. Given the number of frames, n , the time complexity of the subsequence comparison process is $O(n \cdot (n+1)/2)$. Methods using the same framework include [4, 6], which, respectively, use structural similarity (SSIM) [6] and histogram correlation to measure the similarity between subsequences.

Being different from [8], in [5, 7], the authors lexicographically sorted the features (Tamura texture and local binary pattern are used as features in [5, 7], respectively) corresponding to each frame; then, neighbouring features which are close enough to each other in the feature space correspond to duplicated frames. In this manner, the time cost spent on identifying matching frames is theoretically reduced to $O(k \cdot n \cdot \log n)$, where k is the feature dimension. It should be noted that, for lexicographical sorting, the features corresponding to the frames of the entire video have to be simultaneously stored in the memory. From an

implementation perspective, in memory-constrained environments, with the increase of k and n , the storage needed by lexicographical sorting can sometimes exceed the memory capacity. In such cases, the features have to be stored on the disk instead; then, the sorting procedure will involve frequent disk access which is rather slow.

Taking the characteristic of the encoding process into consideration, Subramanyam and Emmanuel [10] extracted the histogram of oriented gradients (HOG) features from block pairs colocated in neighbouring I, B and P, B frame pairs; then, high correlation between the HOG features discloses duplicated blocks and hence frames. However, this method can only be used to detect the case that the source and target frames are placed next to each other, which is quite uncommon.

The methods discussed above unexceptionally detect the duplication behaviour by a fixed global threshold, which makes them less robust. Although some features can be, to some extent, robust to degradation, it is obvious that a threshold calibrated for a certain condition may not be suitable for others. This problem becomes particularly noticeable when processing degraded videos since there are too many unstable factors caused by, e.g., compression artifact or manually added noise.

One subject closely associated with FDD is near-duplicate identification (NDI) [11–14], whose major concern is the copyright issues. Unlike FDD, in NDI, the video clip in query is known, while the goal of FDD is to find all duplicated frame pairs within the video sequence where each frame can be possibly forged; therefore, FDD is more challenging in terms of time complexity, which increases quadratically with the number of frames [11]. Another major difference between FDD and NDI is that, in NDI, the potential attacks can be much stronger than in FDD, and the pirated videos can be geometrically transformed (e.g., picture in picture or recaptured videos) or inserted into logos or subtitles. In this sense, methods for NDI should be more robust while less discriminative than FDD.

3. Proposed Method

3.1. The Pipeline. We detect frame duplication in a coarse-to-fine manner. The pipeline consists of two key steps: coarse matching and duplication verification, as shown in Figure 2. The main concern of the coarse matching stage is to significantly reduce the burden of computation for the second stage. Given an input video V , following [8], we firstly divide V into $T - L + 1$ overlapping subsequences, where T is the total number of frames and L is the subsequence length. For each subsequence t ($1 \leq t \leq T - L + 1$), using LSH, we identify from t 's successive subsequences the ones that are visually similar to t . In this way, we cluster subsequence t and its duplication candidates into the same group. Then, in the duplication verification stage, we perform image registration between each pair of corresponding frames, respectively, in t and the duplication candidates. Through image registration, we obtain a series of offset fields, and the zero-offset fields verify duplications.

FIGURE 2: The pipeline of the proposed method. The pipeline consists of two key steps: coarse matching by LSH and duplication verification. This diagram depicts the process of identifying the duplication of a given subsequence t . Please see the text for details.

3.2. Coarse Matching by LSH. The left part of Figure 2 depicts the process of coarse matching for a given subsequence t . For subsequence t , we would like to find out, from subsequence $t + L$ to $T - L + 1$, the duplication candidates $c_{t,1}, c_{t,2}, \dots, c_{t,k}$ (where $k \leq T - t - 2 \cdot (L - 1)$), which are perceptually similar to t , such that the duplication verification procedure, which is more accurate but slower, can compare t only with $c_{t,i}$ ($1 \leq i \leq k$) instead of with all the succeeding subsequences of t . To this end, what we need is a feature which is sensitive to content (i.e., objects, objects' shape, and objects' position) change while robust against image degradation. Although many image hashing schemes can be used for this purpose (e.g., the wavelet-based [15] and SVD-based [16]), we find that a block-wise GIST [17] feature meets our requirement best in terms of the tradeoff between robustness, discriminative power, and computational time. For each frame, we extract the block-wise GIST descriptor; then, the descriptors extracted from the L frames in the subsequences t and $t + L + i$ ($0 \leq i \leq T - L + 1 - t$) are, respectively, concatenated to form one-dimensional features $f_{s,t}, f_{s,t+L+i}$ for the corresponding subsequences.

We exploit LSH to determine whether a feature $f_{s,t+L+i}$ is sufficiently close to $f_{s,t}$. Given an error probability $P_e > 0$ and a distance threshold R , when $\|f_{s,t} - f_{s,t+L+i}\| \leq R$, LSH guarantees that $P_c \geq 1 - P_e$, where P_c is the collision probability for the hash value of $f_{s,t}$ and $f_{s,t+L+i}$. In this paper, we use the p -stable distribution-based LSH [18]:

$$h = \left\lfloor \frac{\mathbf{a} \cdot \mathbf{f} + b}{\omega} \right\rfloor, \quad (1)$$

where $\mathbf{f}$ is the feature vector being hashed, $\mathbf{a}$ is a real-valued vector whose elements are independently drawn from a standard normal distribution $N(0, 1)$, which has been proven to be 2-stable [18] (since we use the l_2 -norm to measure the difference between features), and b is a real scalar uniformly drawn from $[0, \omega]$, where ω is a real scalar.

To produce more reliable results, we construct H hash tables, and subsequence $t + L + i$ is considered as a

duplication candidate of t only when the hash values of $f_{s,t}$ and $f_{s,t+L+i}$ collide more than $H/2$ times.

If the collected duplication candidates include ξ or more consecutive subsequences ($\xi = 10$ in our experiments, i.e., about 0.5 second), then these consecutive subsequences are considered as static scenes and discarded.

Note that this coarse matching stage also involves a distance threshold, R , but this distance threshold is different from that used in the existing works in that R is not used for making the final decision. The coarse matching step is used to eliminate unnecessary computations; therefore, when choosing R , we do not have to consider much about the tradeoff between robustness and distinctiveness; we should just guarantee that the duplication of t is a subset of $\{c_{t,1}, c_{t,2}, \dots, c_{t,k}\}$. In fact, in practice, R is not necessarily explicitly assigned, and we will discuss this in more detail in the Experimental Results section.

At the end of this stage, each subsequence t is associated with a set of potential duplication candidates $\{c_{t,1}, c_{t,2}, \dots, c_{t,k}\}$. The duplication verification step is then performed between t and these candidates.

3.3. Duplication Verification. For each pair of subsequence t and its duplication candidate $c_{t,i}$, we perform image registration between the corresponding frames to check whether the two frames contain the same objects and whether the shape and position of the corresponding objects happen to be identical. If so, the registration will yield zero-valued offset fields. However, it is not easy to stably obtain correct registration results for degraded images. As shown in Figure 1, even the lossy compression itself can result in substantial changes between a frame and its copy, which usually cause registration faults. To solve this problem, we propose to find the stable regions in the frames and rely on these regions more than the unstable areas during the registration procedure. We use a variant of Harris corneriness response proposed in [19] to measure the stability of the local structure of a pixel:

$$M = \frac{AB - C^2}{A + B}, \quad (2)$$

where $A = g * (\partial F / \partial x)$, $B = g * (\partial F / \partial y)$, and $C = g * ((\partial F / \partial x)(\partial F / \partial y))$; F denotes a given frame, g is a two-dimensional Gaussian kernel, and $*$ denotes the convolution operation.

For a frame F , a large value at $M(x, y)$ indicates that both eigenvalues of the autocorrelation matrix corresponding to $F(x, y)$ are large. This means that the signal changes significantly in two orthogonal directions; such points have been shown to be stable under various conditions except for scale change [20, 21]. We use M to weight different regions in a frame during the registration process, and the registration objective can be written as

$$E(\mathbf{v}) = \sum_{x,y} \mathbf{W}_D(x, y) \cdot D(\mathbf{v}(x, y)) + \sum_{\langle(m,n),(x,y)\rangle \in \mathcal{N}} \mathbf{W}_S(x, y) \cdot S(\mathbf{v}(x, y), \mathbf{v}(m, n)), \quad (3)$$

where D is the data term which measures the difference between the local structures around the matching pixels, S is the smoothness term which guarantees that neighbouring pixels have similar offsets, $\mathbf{v}(x, y)$ is the offset for point (x, y) , and $\langle(m, n), (x, y)\rangle$ denotes an edge in the 4-neighbourhood system $\mathcal{N}$.

$\mathbf{W}_D$ and $\mathbf{W}_S$ are weighting matrices such that

$$W_D(x, y) = \begin{cases} 1 + M'(x, y), & \text{if } M'(x, y) > \epsilon, \\ 0, & \text{otherwise,} \end{cases} \quad (4)$$

$$W_S(x, y) = \max(W_D(x, y), \kappa), \quad (5)$$

where $M'(x, y) = \max_{-r \leq a, b \leq r} \{\tilde{M}(x + a, y + b)\}$, with $\tilde{M}$ being the normalized version of M . We use the maximal filtering to diffuse the impact of the stable points to a small range around them.

ϵ in (5) is quite a small value ($1e-5$ in our implementation) used to mask out the excessively smooth regions when computing the data term (3). Although the Harris corneriness response weights the smooth regions less, the areas which are excessively flat still cause troubles during registration. The local structure of such regions can be easily changed by small perturbation. Figure 1(f) is an apparent illustration of this phenomenon, where large bright spots (significant differences between visual word indices) all locate on the wall or the floor, which are both rather smooth. Such excessively smooth regions will result in high data cost which is inconsistent with the real situation. Based on this observation, we set the threshold ϵ to remove the impact of the data costs in those areas. As a consequence, the offset field within those regions is completely controlled by the smoothness term (4); we therefore add a truncation value κ in (6) to guarantee that the smoothness constraint is always above a certain level.

The data term (3) and smoothness term (4) are, respectively, defined as

$$D(\mathbf{v}(x, y)) = \|f_{\text{src}}(x, y) - f_{\text{tar}}(x + \mathbf{v}_x(x, y), y + \mathbf{v}_y(x, y))\|_1, \quad (6)$$

$$S(\mathbf{v}(x, y), \mathbf{v}(m, n)) = \min(\alpha |\mathbf{v}_x(x, y) - \mathbf{v}_x(m, n)|, d) + \min(\alpha |\mathbf{v}_y(x, y) - \mathbf{v}_y(m, n)|, d), \quad (7)$$

where $f_{\text{src}}(x, y)$ is the feature for point (x, y) in the source frame, $f_{\text{tar}}(x + \mathbf{v}_x(x, y), y + \mathbf{v}_y(x, y))$ denotes the feature for $(x + \mathbf{v}_x(x, y), y + \mathbf{v}_y(x, y))$ in the target frame (we use the SIFT descriptor [22] extracted on a single scale as the feature for each pixel), and $\mathbf{v}_x(x, y)$ and $\mathbf{v}_y(x, y)$ denote the offset in horizontal and vertical directions, respectively. We use the truncated l_1 -norm in (8) to account for discontinuities in the offset field, and α is used for balancing the data term (3) and smoothness term (4) (α can also be combined into (5), we write it here for clearness).

We use the dual-layer loopy belief propagation in [23] to minimize the objective function $E(\mathbf{v})$. By decoupling the smoothness term in (4) into two parts in (8) corresponding to two directions, the complexity of message update in each iteration is reduced from $O(nL^4)$ to $O(nL^2)$, where n is the number of pixels in each frame and L is the number of possible offsets in each direction. The complexity is further reduced to $O(nL)$ by the distance transform proposed in [24]. The multigrid message passing scheme in [21] is also exploited to significantly reduce the total number of iterations.

Optical flow (e.g., [25]) or SIFT flow [23] can also be used for image registration, which is intrinsically a pixel-wise correspondence estimation process. However, neither of them can obtain expected results for degraded videos. The difference between our objective and that of optical flow and SIFT flow is obvious: we encode the stability information of different regions into the matching objective, which makes our method quite robust against video degradation. Moreover, in our objective, there is no small displacement term which is used in SIFT flow so that the registration can be more sensitive to subtle changes between two frames. Figure 3 shows two representative examples demonstrating the difference between the three methods. The offset fields are visualized with the color encoding scheme in [26]; please see Figure 4 for more details.

Compared with the typical methods for FDD [4–8], our method relies on image registration rather than the feature extraction and thresholding strategy. Conforming to the data similarity and smoothness constraints, the correspondences between the pixels of two frames are established in an “optimal” manner through a probabilistic inference process (i.e., the dual-layer loopy belief propagation); furthermore, the pixels with high Harris corner response are usually located on the boundaries of objects; therefore, when Harris corner response is integrated into the registration objective, the registration can be, to some extent, considered

FIGURE 3: The comparison between the three methods for image alignment. Top row: (a, b) a frame and its copy. Bottom row: (a, b) two consecutive frames. The video is recorded by using a still camera. The person in a black coat is walking toward left, and the person's gesture slightly changed across the two frames. (c)–(e) In both rows, the offset fields calculated based on optical flow [25] (we use the implementation in [27]), SIFT flow [23], and our method, respectively. Despite the impressive result for the motion of the walking person, optical flow is too sensitive to perturbations caused by lossy compression; it gets nonzero offsets all over the fields for both cases, particularly for the wall region in the first row. SIFT flow obtains better result than optical flow for the first case, but there is still a large area of nonzero values corresponding to the smooth wall. On the contrary, SIFT flow is not sensitive enough to subtle changes: it fails to detect the motion of the person in the second case. In contrast, our method correctly calculates both offset fields.

as an object-level matching process. As a result, even though the registration objective involves more parameters (i.e., κ , α , and d) than typical feature extraction and thresholding-based methods do, we will show in our experimental part that once the parameters are calibrated, the proposed method can perform more robustly than the feature extraction and thresholding-based methods.

4. Experimental Results

4.1. The Dataset. As far as we know, there is no publicly available dataset dedicated for FDD evaluation. Therefore, we created a dataset to evaluate the performance of the proposed method, especially for the degraded cases. We captured five indoor and eight outdoor video clips (named “v01” to “v13,” and “v01”~“v05” are indoor scenes) with Panasonic HDC-Z10000GK camcorder. The videos were shot in the Science Park of Harbin Institute of Technology. The clips are captured from different scenes, and their contents include characters, landscapes, buildings, and plants. Several screenshots from our dataset are shown in Figure 5. The video clips are H.264 encoded by the built-in codec, and then we convert these clips into the .mp4 format with Adobe Premiere Pro CS 5.5. The resolution of the clips is 1920×1080 , and the frame rate is 25 FPS. Based on these original clips, we created three forgery subsets: the MCOMP subset, the MCOMP + AGN subset, and the MCOMP + INT subset. The details of these subsets are listed in Table 1. Each original clip corresponds to 9 forged versions, and the whole dataset consists of 117 forged videos in all.

The magnitude of the additive Gaussian noise and intensity change is moderate so that they are hardly perceptible. The duration of the forged video clips varies from 8 seconds to 30 seconds.

4.2. The Efficiency of LSH-Based Coarse Matching. As mentioned above, given a subsequence t , in the coarse matching stage, we exploit LSH to find C , with C being the set of duplication candidates of t . In theory, to use the

p -stable distribution-based LSH, we have to assign the distance threshold R and the error probability P_e to determine the parameter ω in (1). However, since we only use LSH as a coarse matcher and the result of the coarse matching does not have to be quite accurate, we should just make sure that $d \in C$, where d is the duplication of t . In this sense, we can determine ω by training rather than by firstly assigning R and P_e and then calculating ω from R and P_e .

To make sure $d \in C$, we define ϕ as the completeness of the correctly collected duplication candidates, and for the training set, we should choose ω such that

$$\phi = \frac{n_c}{n_g} = 1, \quad (8)$$

where n_c is the number of correctly collected duplication candidates and n_g is the actual number of duplicated subsequences. Given the premise in (9), we use the average number of collisions, n_{ave} , to measure the efficiency of the coarse matching step as follows:

$$n_{ave} = \frac{n_t}{T - L + 1}, \quad (9)$$

where n_t is the total number of collisions. It is straightforward that n_{ave} monotonically increases with ω , and we prefer smaller n_{ave} when (9) is guaranteed.

We randomly select four video clips which are most seriously degraded (i.e., have been attacked by additive Gaussian noise of the standard deviation of 10 or by downscaling the intensity by 3%) from the MCOMP + AGN and MCOMP + INT subset, respectively, to train the parameter ω . The block size of the block-wise GIST feature is 25×25 , the subsequence length L is set to 5, and we scale each frame to 12.5% of their original size before feature extraction. We constructed 80 hash tables for the coarse matching stage; therefore, a pair of subsequences is considered to be identical only when their hash values collide more than 40 times. Under this configuration, ϕ and n_{ave} vary with ω as shown in Figure 6. We set $\omega = 0.45$, where ϕ reaches 1, and $n_{ave} \approx 0.094$.

FIGURE 4: The color encoding of offset fields. (a) The orientation and magnitude of the offsets are represented by the hue and saturation of the corresponding pixel. (b) The sketch map illustrating the offset vectors relevant to different colors in (a).

FIGURE 5: Sample pictures captured from the clips in our dataset. (a, i) Indoor scenes. (b)–(h) Outdoor scenes.

It is obvious that the value of ϕ and n_{ave} depends on the content of the videos. We list in Table 2 the mean values of ϕ and n_{ave} of the 9 forged versions corresponding to each

original clip. Most of the mean values of ϕ are above 0.98. The two smaller values (0.94 for v03 and v13) are both caused by intensity degradation. For video clip v03 in group

TABLE 1: The description of the three forgery subsets.

Subset	Description
MCOMP	MCOMP stands for multiple compression. For each original clip, we use MATLAB R2014a to randomly copy k ($5 \leq k \leq 35$) consecutive frames and paste them to another random position in the timeline and resave the clips with the quality factors of 100, 80, and 60, respectively. Finally, each clip has been compressed three times (by camcorder, Adobe Premiere, and MATLAB, respectively). We denote the three levels of compression as MCOMP100, MCOMP80, and MCOMP60 groups, respectively.
MCOMP + AGN	The same steps as the MCOMP100 group are performed. Besides, before resaving the forged clips, additive Gaussian noise with the standard deviations of 1, 5, and 10 is added to the target frames, i.e., each original clip corresponds to three forged clips subject to different levels of additive noise. For simplicity, in the rest of this paper, we refer to these three levels of forgeries as AGN1, AGN5, and AGN10 groups, respectively.
MCOMP + INT	The same steps as the MCOMP100 group are performed. Besides, before resaving the forged clips, the intensity of the pixels in the target frames is downscaled to 99%, 97%, and 95% of their original values. In the following, we refer to these three levels of forgeries as INT99, INT97, and INT95 groups, respectively.

FIGURE 6: ϕ and n_{ave} increase as ω gets larger. ϕ reaches 1 when $\omega = 0.45$.

INT95, the value of ϕ was only 0.61. This is because the intensity of the pixels of the duplicated frames is large; therefore, a small scaling factor can result in remarkable intensity change. On average, 2% duplicated subsequences are missed during the coarse matching stage.

On the contrary, the average of mean n_{ave} for different scenes is 0.10; this implies that, in our duplication verification stage, on average, we just need to perform 0.1 comparison for each subsequence. In contrast, without the coarse matching stage, on average, we have to compare each subsequence with about $T/2$ other subsequences, where T is the number of frames. T is typically larger than 200 in our dataset and can be much larger in practice. In this sense, for our dataset, the computational load for the duplication verification stage is reduced by 3 orders of magnitude. We will show later that it is worth performing this coarse matching step, in spite of its $O(T \cdot (T+1)/2)$ time complexity.

4.3. The Detection Capability. In this section, we investigate the detection capability of our method. In our implementation, we randomly select 7 forged clips to calibrate κ , α , and d , and κ , α , and d are empirically set to be 1.8, 635, and 12,800, respectively. The frames are resized by a factor of 25% (for acceleration) before registration. We binarized the resulted offset field images and masked out the nonzero

regions whose area is less than 0.1% of the whole field image to account for outliers. We use precision (10), recall (11), and F_1 -score (12) to evaluate the performance:

$$\text{precision} = \frac{TP}{TP + FP}, \quad (10)$$

$$\text{recall} = \frac{TP}{TP + FN}, \quad (11)$$

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \quad (12)$$

where TP, FP, and FN are the number of correctly detected duplication frame pairs, the number of falsely detected duplication frame pairs, and the number of undetected duplication pairs, respectively.

We compared our method with [4, 8] (denoted Farid and Li, respectively), and the related parameters are set to be identical to those in [4, 8], respectively. The comparison results for the MCOMP subset are shown in Tables 3–5. The label “v01” in the first row denotes the forged version of video clip v01 in the current forgery group, and “average” means the average value of v01 to v13, and the same hereinafter. For v07 to v12 in the MCOMP100 group, all three methods performed perfectly. For v05 and v13, our method obtained rather low precision; this is because some shots in v05 and v13 are almost still, and the duplication verification step failed to differentiate the excessively similar frames. On the contrary, [4] is quite effective in terms of discriminative power. Our method outperformed the other two in the first four cases. It should be noted that [8] detected none of the duplicated frames in v02 to v05 and v13 for differing reasons. In [8], when the temporal correlations of the frames in a subsequence are all above a certain value, this subsequence is considered as static and then discarded; therefore, in v05 and v13, the subsequences whose frames are quite similar to each other have not been compared with other subsequences at all. In contrast, the duplicated frames in v02 to v04 are missed due to the inappropriate correlation threshold.

For the MCOMP80 group, the precision of [4] for v05 and v13 dropped to 0.44 and 0.50, respectively, and the performance of [4] for the rest of the clips just slightly changed. The performance of [8] was rather poor for this

TABLE 2: The mean values of ϕ and n_{ave} of the forged versions corresponding to each original clip.

	v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Mean $_{\phi}$	0.99	0.98	0.94	1.00	0.99	1.00	0.96	0.99	0.98	1.00	1.00	1.00	0.94	0.98
Mean $_{n_{\text{ave}}}$	0.38	0.04	0.08	0.10	0.34	0.10	0.03	0.06	0.03	0.05	0.02	0.02	0.07	0.10

TABLE 3: The comparison results for the MCOMP100 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.22	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.21	0.88
	Farid [8]	1.00	0.00	0.00	0.00	0.00	0.00	0.74	1.00	1.00	1.00	1.00	1.00	0.00	0.60
	Li [4]	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.85
Recall	Ours	0.97	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.88	0.99
	Farid [8]	0.17	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.55
	Li [4]	0.00	0.00	0.30	0.70	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.77
F_1 -score	Ours	0.98	1.00	1.00	1.00	0.36	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.34	0.90
	Farid [8]	0.29	0.00	0.00	0.00	0.00	0.85	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.55
	Li [4]	0.00	0.00	0.46	0.82	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.79

TABLE 4: The comparison results for the MCOMP80 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.12	1.00	1.00	0.50	1.00	1.00	1.00	1.00	0.09	0.82
	Farid [8]	1.00	0.00	0.00	0.00	0.00	0.82	1.00	0.00	1.00	0.54	0.00	0.00	0.00	0.34
	Li [4]	0.00	0.00	1.00	1.00	0.44	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.50	0.76
Recall	Ours	0.97	1.00	1.00	1.00	1.00	1.00	0.83	1.00	1.00	1.00	1.00	1.00	1.00	0.98
	Farid [8]	0.38	0.00	0.00	0.00	0.00	1.00	1.00	0.00	1.00	0.91	0.00	0.00	0.00	0.33
	Li [4]	0.00	0.00	1.00	0.87	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.83
F_1 -score	Ours	0.98	1.00	1.00	1.00	0.22	1.00	0.91	0.67	1.00	1.00	1.00	1.00	0.17	0.84
	Farid [8]	0.55	0.00	0.00	0.00	0.00	0.90	1.00	0.00	1.00	0.68	0.00	0.00	0.00	0.32
	Li [4]	0.00	0.00	1.00	0.93	0.61	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.65	0.78

TABLE 5: The comparison results for the MCOMP60 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.16	1.00	1.00	1.00	1.00	1.00	0.50	1.00	0.37	0.85
	Farid [8]	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	0.38
	Li [4]	0.00	0.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	1.00	0.54
Recall	Ours	0.87	1.00	1.00	1.00	1.00	1.00	1.00	0.75	0.83	1.00	0.50	1.00	0.96	0.92
	Farid [8]	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	0.38
	Li [4]	0.00	0.00	0.48	0.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	0.96	0.50
F_1 -score	Ours	0.93	1.00	1.00	1.00	0.28	1.00	1.00	0.86	0.91	1.00	0.50	1.00	0.53	0.85
	Farid [8]	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	0.38
	Li [4]	0.00	0.00	0.65	0.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	0.98	0.51

group: it failed to detect 8 out of 13 forged video clips. In this group, the precision for v08 and recall for v07 of our method decreased to 0.50 and 0.83, respectively, and the results for other clips were almost the same with those in the MCOMP100 group.

When the quality factor of the last compression dropped to 60, the performance of [4] significantly decreased. Specifically, 6 video clips were mistakenly judged as unchanged. In contrast, most results of our method maintained a relatively stable value.

The detection results for the MCOMP + AGN subset are shown in Tables 6–8. In the AGN1 group (Table 6), our method outperformed the other two for the first four

cases. Our recall rate for v08 is a little lower than the other two. Besides, for v05 and v13, [4] still obtained better results than our method, while the precision for v05 dramatically dropped to less than 0.50. [8] obtained the worst results for v01 to v03, v05, v06, and v13. In Table 7, it is interesting that the precision of [4] for v05 in the AGN5 group is 1.00, which was supposed to be less than 0.50 due to the stronger noise. According to our observation, the difference between the source and target frames in v05 in this group is indeed small, which may be caused by the encoding mechanism of the lossy compression. In Table 8, for most cases, the precision and recall of [4, 8] dropped to 0. Contrarily, our method is

TABLE 6: The comparison results for the AGN1 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.07	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.87
	Farid [8]	0.00	0.00	0.00	1.00	0.00	0.62	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.59
	Li [4]	0.00	0.00	1.00	1.00	0.46	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.80
Recall	Ours	0.92	1.00	1.00	1.00	1.00	1.00	1.00	0.97	1.00	1.00	1.00	1.00	1.00	0.99
	Farid [8]	0.00	0.00	0.00	0.56	0.00	1.00	1.00	1.00	1.00	1.00	1.00	0.79	0.00	0.57
	Li [4]	0.00	0.00	0.86	0.53	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.80
F_1 -score	Ours	0.96	1.00	1.00	1.00	0.14	1.00	1.00	0.98	1.00	1.00	1.00	1.00	0.29	0.87
	Farid [8]	0.00	0.00	0.00	0.72	0.00	0.76	1.00	1.00	1.00	1.00	1.00	0.88	0.00	0.57
	Li [4]	0.00	0.00	0.92	0.69	0.63	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.79

TABLE 7: The comparison results for the AGN5 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.22	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.87
	Farid [8]	1.00	0.00	0.00	0.00	0.00	1.00	1.00	0.37	0.71	0.59	1.00	1.00	0.00	0.51
	Li [4]	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.85
Recall	Ours	1.00	1.00	1.00	1.00	0.97	1.00	0.97	0.88	1.00	1.00	1.00	1.00	0.86	0.98
	Farid [8]	0.80	0.00	0.00	0.00	0.00	1.00	0.74	1.00	1.00	0.76	1.00	0.22	0.00	0.50
	Li [4]	0.00	0.00	1.00	0.35	1.00	1.00	0.35	1.00	1.00	1.00	1.00	1.00	1.00	0.75
F_1 -score	Ours	1.00	1.00	1.00	1.00	0.36	1.00	0.99	0.94	1.00	1.00	1.00	1.00	0.22	0.88
	Farid [8]	0.89	0.00	0.00	0.00	0.00	1.00	0.85	0.54	0.83	0.66	1.00	0.36	0.00	0.47
	Li [4]	0.00	0.00	1.00	0.52	1.00	1.00	0.52	1.00	1.00	1.00	1.00	1.00	1.00	0.77

TABLE 8: The comparison results for the AGN10 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.10	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.08	0.86
	Farid [8]	1.00	0.00	0.00	1.00	0.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	0.00	0.54
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	1.00	1.00	0.00	0.23
Recall	Ours	1.00	1.00	1.00	1.00	1.00	1.00	0.70	1.00	1.00	1.00	1.00	1.00	0.88	0.97
	Farid [8]	0.94	0.00	0.00	0.33	0.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	0.00	0.48
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.83	0.00	0.00	1.00	1.00	0.00	0.22
F_1 -score	Ours	1.00	1.00	1.00	1.00	0.18	1.00	0.82	1.00	1.00	1.00	1.00	1.00	0.15	0.86
	Farid [8]	0.97	0.00	0.00	0.50	0.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	0.00	0.50
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.91	0.00	0.00	1.00	1.00	0.00	0.22

rather stable across different groups, and the performance hardly changed.

The detection results for the MCOMP + INT subset are shown in Tables 9–11. Except for v05 and v13, our method is on par with or excelled the other two methods for most cases. Since [8] used correlation of pixel intensity as their feature, which is rather robust against intensity change, in this subset, [8] performed better than it did in the COMP + AGN subset. On the contrary, [4] is rather sensitive to intensity change. For the INT95 group, [4] even detected none duplicated frames at all for 10 out of 13 forged clips.

When comparing frames in degraded videos, the distances between the features can easily fall out of the range of a fixed threshold. As can be seen from Tables 3–11, when the degradation gets stronger, more than half of the forged video clips have been falsely judged as innocent by [4, 8]: none of the duplicated frames has been detected. By contrast, our method performed stably across different test groups. Although, occasionally, our method performed worse than the

other two methods (especially for v05 and v13), the average of the precision, recall, and F_1 -score, without exception, significantly outperformed those of [4, 8]. Even for the strongest degradation groups, i.e., MCOMP60, MCOMP + AGN10, and MCOMP + INT95, the average values of precision, recall, and F_1 -score of our method are above 0.8 (for the worst case, $F_1 = 0.81$ in INT95).

4.4. The Running Time. The running time of the three methods is closely related to the content of the videos. When comparing subsequences, once any corresponding pair of frames is found to be not identical, the comparison between the current pair of subsequences terminates. Therefore, the video clips whose frames are highly similar to each other will result in more processing time. The comparison between the running time of the three methods is given in Table 12. The frames are scaled by a factor of 25% for all the three methods, and all the experiments are conducted on a workstation with

TABLE 9: The comparison results for the INT99 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.09	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.09	0.86
	Farid [8]	1.00	0.00	1.00	1.00	0.00	0.66	1.00	1.00	1.00	0.00	1.00	1.00	0.00	0.67
	Li [4]	0.00	0.00	1.00	0.00	0.34	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.72
Recall	Ours	0.94	0.83	0.83	1.00	1.00	1.00	0.93	1.00	1.00	1.00	1.00	1.00	0.89	0.95
	Farid [8]	0.16	0.00	0.83	1.00	0.00	0.94	0.75	0.36	1.00	0.00	1.00	0.52	0.00	0.50
	Li [4]	0.00	0.00	0.97	0.00	1.00	1.00	0.71	1.00	1.00	1.00	1.00	1.00	1.00	0.74
F_1 -score	Ours	0.97	0.91	0.91	1.00	0.17	1.00	0.96	1.00	1.00	1.00	1.00	1.00	0.17	0.85
	Farid [8]	0.28	0.00	0.91	1.00	0.00	0.78	0.86	0.53	1.00	0.00	1.00	0.68	0.00	0.54
	Li [4]	0.00	0.00	0.98	0.00	0.51	1.00	0.83	1.00	1.00	1.00	1.00	1.00	1.00	0.72

TABLE 10: The comparison results for the INT97 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.19	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.15	0.87
	Farid [8]	0.00	0.00	0.00	1.00	0.00	0.35	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.57
	Li [4]	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.69
Recall	Ours	0.83	1.00	1.00	1.00	0.96	1.00	1.00	0.82	1.00	1.00	1.00	1.00	0.96	0.97
	Farid [8]	0.00	0.00	0.00	0.32	0.00	1.00	1.00	1.00	1.00	1.00	1.00	0.95	0.00	0.56
	Li [4]	0.00	0.00	0.00	0.00	1.00	1.00	1.00	0.45	1.00	1.00	1.00	1.00	0.83	0.64
F_1 -score	Ours	0.91	1.00	1.00	1.00	0.32	1.00	1.00	0.90	1.00	1.00	1.00	1.00	0.26	0.88
	Farid [8]	0.00	0.00	0.00	0.48	0.00	0.52	1.00	1.00	1.00	1.00	1.00	0.96	0.00	0.54
	Li [4]	0.00	0.00	0.00	0.00	1.00	1.00	1.00	0.63	1.00	1.00	1.00	1.00	0.91	0.66

TABLE 11: The comparison results for the INT95 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.16	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.25	0.88
	Farid [8]	0.00	0.00	0.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.62
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	1.00	1.00	0.00	0.23
Recall	Ours	0.54	1.00	0.61	1.00	1.00	1.00	1.00	0.23	1.00	1.00	1.00	1.00	0.86	0.86
	Farid [8]	0.00	0.00	0.00	0.41	0.00	1.00	1.00	1.00	1.00	0.73	1.00	1.00	0.00	0.55
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.33	0.00	0.00	0.00	1.00	0.97	0.00	0.18
F_1 -score	Ours	0.70	1.00	0.76	1.00	0.27	1.00	1.00	0.38	1.00	1.00	1.00	1.00	0.38	0.81
	Farid [8]	0.00	0.00	0.00	0.58	0.00	1.00	1.00	1.00	1.00	0.84	1.00	1.00	0.00	0.57
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	1.00	0.99	0.00	0.19

TABLE 12: The averaged running time (seconds) of each scene of the three methods.

Scenes	Average duration (seconds)	Average running time		
		Ours	Farid [8]	Li [4]
v01	10	447	438	498
v02	11	214	996	567
v03	8	195	228	324
v04	8	218	162	375
v05	17	1979	150	1789
v06	9	263	535	450
v07	19	383	752	1900
v08	9	226	438	443
v09	13	258	394	860
v10	14	293	655	1010
v11	28	461	5226	3932
v12	30	548	242	4732
v13	29	951	169	4020

TABLE 13: The average running time (seconds) of each step in our method.

Steps	Average running time
Frame extraction	32
Coarse matching	209
Duplication verification	252

Intel Core i7-2600 processor and 24 GB RAM. We implemented the three methods with MATLAB R2014a.

As mentioned earlier, in [8], if the correlation coefficients between the frames within a given subsequence are all above a predefined value, the subsequence will be considered as static and discarded. Such subsequences will not be compared to other subsequences, and that is why the running time of [8] is so short for scenes v05 and v13 (recall that [8] detected none of the forgeries corresponding to these two scenes).

In fact, both the structural similarity and correlation coefficients used in [4, 8], respectively, can be computed much faster than the image registration process in our method. For each single pair of frames to be compared, the structural similarity and correlation coefficients can be calculated in about 0.03 and 0.05 second, respectively. In contrast, in our method, the image registration procedure takes about 4 seconds for each pair of frames to be compared. Even so, our method is faster than the other two for most cases. The coarse matching step plays an important role for acceleration. As we have demonstrated in Table 2, the total number of subsequences which need finer duplication verification can be reduced by several orders of magnitude, especially for the video clips whose content changes rapidly across frames. The average running time of each step in our method is listed in Table 13.

The coarse matching step accounts for about 42% of the total running time, and it takes less than one second for each frame on average. Such a step is well worth performing: without coarse matching, even a 10-second video clip will cost us several hours to detect the forgery.

5. Conclusion and Future Work

In this paper, we proposed a new method for frame duplication detection, particularly for the degraded videos. Our method detects duplication forgeries in a coarse-to-fine manner and consists of two steps: coarse matching and duplication verification. In the coarse matching stage, we use locality-sensitive hashing to precluster the visually similar subsequences. Through coarse matching, the total number of subsequences which need finer duplication verification can be reduced by several orders of magnitude. The duplication verification step exploits image registration to identify the identical subsequences. We encode the stability information of different regions into the registration objective function such that the registration can work stably for degraded videos. Being different from existing methods, our detection process does not rely on a fixed distance threshold, which is typically unreliable for degraded videos. Experimental results show that our method outperformed state-of-the-art

methods for most cases and exhibited outstanding robustness under different conditions. However, our method cannot distinguish between highly similar frames; as a result, for the video clips whose content just slightly changes across frames, the precision can be rather low. Further efforts should be made to improve the discriminative power of the registration process.

Data Availability

The dataset used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61771168) and the Natural Science Foundation of Heilongjiang Province of China (Grant no. F2017014).

References

- [1] Z. Huang, L. Fang, and S. Li, "Subpixel-pixel-superpixel guided fusion for hyperspectral anomaly detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 99, pp. 1–10, 2020.
- [2] D. Meng, X. Wang, M. Huang, L. Wan, and B. Zhang, "Robust weighted subspace fitting for DOA estimation via block sparse recovery," *IEEE Communications Letters*, vol. 24, no. 3, pp. 563–567, 2020.
- [3] Y. Wang, L. Wang, C. Yu et al., "Constrained-target band selection for multiple-target detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 8, pp. 6079–6103, 2019.
- [4] F. Li and T. Huang, "Video copy-move forgery detection and localization based on structural similarity," in *Proceedings of the 3rd International Conference on Multimedia Technology (ICMT 2013)*, London, UK, June 2014.
- [5] S. Y. Liao and T.-Q. Huang, "Video copy-move forgery detection and localization based on tamura texture features," in *Proceedings of the 2013 6th International Congress on Image and Signal Processing (CISP)*, Hangzhou, China, December 2013.
- [6] G. S. Lin, J.-F. Chang, and C.-H. Chuang, "Detecting frame duplication based on spatial and temporal analyses," in *Proceedings of the 2011 6th International Conference on Computer Science & Education (ICCSE)*, Singapore, August 2011.
- [7] D. Tralic, S. Grgic, and B. Zovko-Cihlar, "Video frame copy-move forgery detection based on cellular automata and local binary patterns," in *Proceedings of the 2014 X International Symposium on Telecommunications (BIHTEL)*, Sarajevo, Bosnia, October 2014.
- [8] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting duplication," in *Proceedings of the 9th Workshop on Multimedia & Security*, pp. 35–42, Cardiff, Wales, UK, July 2007.

- [9] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [10] A. Subramanyam and S. Emmanuel, "Video forgery detection using hog features and compression properties," in *Proceedings of the 2012 IEEE 14th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 89–94, Banff, AB, Canada, September 2012.
- [11] A. Joly, O. Buisson, and C. Frelicot, "Content-based copy retrieval using distortion-based probabilistic similarity search," *IEEE Transactions on Multimedia*, vol. 9, no. 2, pp. 293–306, 2007.
- [12] M. Douze, H. Jegou, and C. Schmid, "An image-based approach to video copy detection with spatio-temporal post-filtering," *IEEE Transactions on Multimedia*, vol. 12, no. 4, pp. 257–266, 2010.
- [13] D. Q. Zhang and S.-F. Chang, "Detecting image near-duplicate by stochastic attributed relational graph matching with learning," in *Proceedings of the 12th Annual ACM International Conference on Multimedia*, pp. 877–884, New York, NY, USA, October 2004.
- [14] O. Chum, J. Philbin, M. Isard, and A. Zisserman, "Scalable near identical image and shot detection," in *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, pp. 549–556, Amsterdam, The Netherlands, 2007.
- [15] R. Venkatesan, S.-M. Koon, M. H. Jakubowski, and P. Moulin, "Robust image hashing," in *Proceedings of the 2000 International Conference on Image Processing*, pp. 664–666, Vancouver, Canada, 2000.
- [16] S. S. Kozat, R. Venkatesan, and M. K. Mihcak, "Robust perceptual image hashing via matrix invariants," in *Proceedings of the 2004 International Conference on Image Processing (ICIP'04)*, pp. 3443–3446, Singapore, October 2004.
- [17] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.
- [18] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p -stable distributions," in *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, pp. 253–262, Brooklyn, NY, USA, 2004.
- [19] J. A. Noble, "Finding corners," *Image and Vision Computing*, vol. 6, no. 2, pp. 121–128, 1988.
- [20] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Alvey Vision Conference*, Manchester, UK, September 1988.
- [21] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 151–172, 2000.
- [22] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pp. 1150–1157, Kerkyra, Greece, September 1999.
- [23] C. Liu, J. Yuen, and A. Torralba, "Sift flow: dense correspondence across scenes and its applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 978–994, 2011.
- [24] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *International Journal of Computer Vision*, vol. 70, no. 1, pp. 41–54, 2006.
- [25] A. Bruhn, J. Weickert, and C. Schnorr, "Lucas/kanade meets horn/schunck: combining local and global optic flow methods," *International Journal of Computer Vision*, vol. 61, no. 3, pp. 211–231, 2005.
- [26] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.
- [27] C. Liu, *Beyond pixels: exploring new representations and applications for motion analysis*, Ph.D. dissertation, 2009.