Perceptual Video Hashing With Secure Anti-Noise Model for Social Video Retrieval

Lv Chen, Dengpan Ye, and Yueyun Shang

***Abstract*—In real scenarios, videos are usually corrupted by multiple types of noise, which brings great challenges to retrieving social videos. However, most of the current video hashing methods for video retrieval consider the attack of a single noise model, and rarely discuss when dealing with complex noise models, which is not conducive to solving the above difficulties. Thus, we describe a novel video hashing with secure anti-noise model (SANM). To improve the robustness of noise attacks, the input video is reconstructed into an SANM by low-rank representation (LRR) and random subspace partition (RSP). LRR is useful technique for capturing the global structure of data. It focuses on recovering the underlying subspace in noisy environment and helps to make the proposed model robust to multiple noises. In addition, using chaotic mapping to control the generation of RSP can ensure the security of proposed model. Then, a new subspace decomposition descriptor (SDD) is proposed. SDD is obtained by calculating the invariant distances of the factor matrices obtained by Tucker decomposition, and is used to decompose SANM to derive a compact hash. Various experiments demonstrate that the SANM hashing performs better than several state-of-the-art algorithms in terms of good robustness and discrimination, and it can accurately retrieve social videos.**

***Index Terms*—Low-rank representation (LRR), perceptual hashing, privacy protection, video retrieval.**

I. Introduction

WITH the quick advancement of social network platforms and multimedia technologies, social video has progressively taken over as the primary means of information transmission. When videos are uploaded, downloaded, forwarded, and collected on social platforms, social videos are often modified and redistributed, thus, causing problems of content identification in large-scale databases. Social network videos are strictly from complicated sources, and as they spread over the network, they may be attacked by content-preserving procedures, such as multiple noise models, independent/independent identically distributed, rotated, cropped, etc. It brings great challenge to video retrieval on social network.

Perceptive video hashing, fortunately, can solve this problem by associating multimedia unique hashing sequences. A developing area of study in multimedia and information security is perceptual video hashing. In order to provide a concise hash sequence that serves as a summary of the perceptual content, it compresses the internal features derived from video content. Video hashing is now widely used in video (retrieval, recognition, tracking, and copy detection) and a variety of other applications [1], [2], [3], [4]. Some commonly used symbols and important properties are defined and described below to help with the introduction of video hashing.

Searching for a hash function $H(\mathbf{V})$ that converts a video $\mathbf{V}$ into the brief sequence $\mathbf{h}=H(\mathbf{V})$ is the essence of perceptual video hashing. $\mathbf{V}$ stands for the input video, while $\mathbf{V}_1$ and $\mathbf{V}_2$ are the visually similar and visually distinct versions of $\mathbf{V}$, respectively. In general, perceptive video hashing need to satisfy two basic properties: 1) robustness and 2) discrimination. In particular, in order to avoid forgery during hash generation, video hashing algorithm should be key dependency for some security applications (e.g., authentication and forensics) [5], [6]. Robustness, discrimination, and key dependency are defined as follows.

Robustness: The robustness of the video hash function says that it produces the same hash sequence for perceptual digital videos, or something extremely close to it

$$P\{r[H(\mathbf{V}), H(\mathbf{V}_1)] < T\} \geq 1 - \varepsilon \tag{1}$$

where $P(\cdot)$ and $r(\cdot,\cdot)$ denote the likelihood that an event will occur and the metric function used to compare the hashes of two videos, respectively. T is a predetermined threshold, it ought to be near zero ($0 < \varepsilon < 1$).

Discrimination: Different hashes should be generated to represent visually distinct films, as stated as follows:

$$P\{r[H(\mathbf{V}), H(\mathbf{V}_2)] \geq T\} \geq 1 - \varepsilon. \tag{2}$$

Key Dependency: Refers to using a different secret key to encrypt the video and produce a different video hash

$$P\{r[H(\mathbf{V}, k_1), H(\mathbf{V}, k_2)] \geq T\} \geq 1 - \varepsilon \tag{3}$$

where $H(\cdot,\cdot)$ denotes the hash function with a secret key-based encryption system and two distinct keys produced at random are k_1 and k_2.

Perceptual video hashing has seen continual innovation and expansion in new applications during the past ten years

Manuscript received 21 February 2023; revised 29 April 2023 and 22 May 2023; accepted 2 July 2023. Date of publication 17 July 2023; date of current version 8 January 2024. This work was supported by the National Natural Science Foundation of China NSFC under Grant 62072343. *(Corresponding author: Dengpan Ye.)*

Lv Chen and Dengpan Ye are with the Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China (e-mail: lv.chen@whu.edu.cn; yedp@whu.edu.cn).

Yueyun Shang is with the School of Mathematics and Statistics, South Central University for Nationalities, Wuhan 430070, China (e-mail: 36214001@qq.com).

Digital Object Identifier 10.1109/JIOT.2023.3293609

because to the unceasing efforts of researchers. Since feature extraction is a crucial procedure for the generation of video hashes, a comprehensive overview of the most popular video hashing algorithms can be found below.

1) Spatial-Feature Based Video Hashing Methods: To develop video hashing algorithms, the majority of early research exploited spatial features. These algorithms typically take the important frames of a video and extract the content-based feature from them. Because of this, the majority of algorithms perform well against in-frame actions like frame scaling, noise assaults, and MPEG compression. It is, however, vulnerable to interframe operations (e.g., frame discarding, frame interpolation, and frame shifting). For instance, Mucedero et al. [7] previously described work that extracted the block variance (BV) of video frames to generate hash. Despite being resistant to MPEG compression, BV hashing is susceptible to temporal-based attacks like frame shifting. Then, Coskun and Sankur [8] carried out 3-D discrete cosine transform (3D-DCT) on content-based features of the original video, and quantized the subband of DCT coefficients into binary code. The 3D-DCT hashing is robust blur and MPEG-4 operation, but is fragile to video rotation and cropping. Lee and Yoo [9] derived the video hash by using the gradient direction centroid (GDC) of the block on the frame. The GDC hashing has noise attack performance and applied to video identification, but it is easily affected by geometric transformation (frame shifting and rotation). Nie et al. [10] selects key frames from video chromatography, and uses isometric feature mapping (IFM) of key frames to make hash. The IFM hashing overcomes random frame dropping in video to some extent, this method is easy to be disturbed by noise attack. In a different paper, Yang et al. [11] suggested combining the speeded up robust feature (SURF) with the ordinal measure (OM) to create a novel hash that can withstand a variety of operations, including MPEG compression and video rotation. Additionally, the SURF–OM hashing was previously used to detect video copies, but its discrimination power needs to be increased.

2) Spatial–Temporal-Features-Based Video Hashing Methods: These methods consider both intraframe and interframe to generate video hashing. Therefore, most algorithms can maintain a certain degree of interframe operation while preserving a good resistance to intraframe operation, but they are vulnerable to geometric attacks. In addition, computational complexity of spatial–temporal-features-based video hashing methods is usually higher than video hashing based on spatial features. For example, Li and Monga [12] exploited a tensor decomposition called low-rank tensor approximation (LRTA) to decompose the original video into a short hash. The LRTA hashing is good robustness with video rotation, JPEG compression and frame rate change, but cropping and discrimination should be improved. Saikia [13] used spatiotemporal low-pass bands of 3-D discrete wavelet transform (3D-DWT) as feature vectors, and then generated the hash by extracting the DCT coefficients of the feature vectors. The DWT–DCT hashing achieve good robustness for the operation of frame dropping and brightness adjustment, but its sensitive to noise attack. Sandeep et al. [14] regarded the input video as a tensor model, performed Tucker decomposition on it, and exploited the factor matrices to derive the hash. This scheme applied for near-identical video indexing and retrieval with good performance and reach good discrimination, but frame shift and video rotation have relatively large room for improvement. By computing gradient edge orientation histogram (EOH) and extracting DCT coefficients, Setyawan and Timotius [15] created a hash. This method generates a hash that is resistant to frame scaling and brightness altering by computing the gradient edge histogram of each frame and choosing the DCT coefficients from the temporal domain. In another work, Nie et al. [16] proposed a video of extracting the feature matrix from the spherical torus (ST). Then, the dimensionality of the feature matrix is reduced by nonnegative matrix factorization (NMF), and the low-dimensional representation of the feature matrix is obtained to generate hash. The ST–NMF hashing is indeed robust to blurring and noise attack, but the discrimination is still less satisfactory. Sandeep et al. [17] used the 3-D radial projection technique (RPT) to extract the feature vector from the video, simplified it through DCT, and encoded it into a short string. The RPT–DCT hashing can resist small angle of rotation and MPEG4 compression, but it is susceptible to watermark embedding. Rameshnath and Bora [18] used temporal wavelet transform (TWT) to extract spatial and temporal features from normalized videos, and used Achlioptas random matrix (ARM) method to reduce the data dimension in transform domain to construct hash. The TWT–ARM hashing is robust to regular frame dropping and interpolation and can be used in near-identical video retrieval. But is fragile to reverse play and watermark embedding.

Recently, Khelifi and Bouridane [19] designed a hash by combining DCT and the discrete sine transform (DST). They calculate the mean value of the nonoverlapping blocks in each frame to form a 3-D array and uses the DCT and DST calibrate signals to derived hashing. The DCT–DST hashing offers good performance for MPEG compression and frame insertion, but is susceptible to the cropping and rotating of videos. Tang et al. [20] divided videos into multiple groups and extracted DCT feature matrix from them. Then, NMF is used to dimensional feature matrix. The DCT–NMF hashing can resist some commonly used operation, such as brightness adjustment and MPEG-2 compression. But is fragile to video rotation and its discrimination needs to be further improved. Chen et al. [21] used ring partition and pipeline histogram to construct a video with rotation and translation invariance model (RTIM), and further decomposed the model to form video hash, which is good robust against video rotation and translation. But, the RTIM hashing is sensitive to interframe operation, such as frame dropping. Tang et al. [22] created a new video hashing algorithm by utilizing 3D-DWT and invariant moments (IMs). In particular, the secondary frame is rebuilt using 3D-DWT, and after that, the IM of the secondary frame is computed to provide a hash. This approach performs well against noise attacks and compression, but it still has to improve in terms of resilience against geometrical attacks. In another work, singular value decomposition (SVD) and DWT were combined to create a novel

TABLE I
VIDEO HASHING ALGORITHM AGAINST COMMON CONTENT-PRESERVING OPERATIONS

Algorithm	Brightness adjustment	White gaussian noise	Mpeg compression	Frame scaling	Video rotation	Frame dropping	Main techniques
[11]	Yes	Yes	Yes	Yes	unknown	No	SURF + OM
[12]	Yes	Yes	Yes	No	Within 5°	unknown	Normalization + PARAFCA
[13]	Yes	Yes	Yes	unknown	unknown	Yes	3D-DWT + DCT
[15]	No	Yes	Yes	Yes	Within 1°	No	EOH + DCT
[17]	Yes	Yes	Yes	Yes	Within 5°	No	3D-RPT + DCT
[18]	Yes	Yes	Yes	unknown	Within 5°	No	TWT + ARM
[19]	Yes	Yes	Yes	No	Within 2°	Yes	DLBM + DCT + DST
[20]	Yes	Yes	Yes	Yes	Within 2°	Yes	2D-DCT + NMF
[21]	Yes	Yes	Yes	Yes	Arbitrary degree	No	RTIM Model + TD
[22]	Yes	Yes	Yes	Yes	Within 1°	Yes	3D-DWT + Invariant Moments
[23]	Yes	Yes	Yes	Yes	Within 2°	Yes	SVD + 2D-DWT
[24]	Yes	Yes	Yes	Yes	Within 2°	No	Sampling RS+ Sampling CCS
[25]	Yes	No	Yes	Yes	Within 2°	Yes	Image hashing + DWT+LLE

video hashing method by Chen et al. [23]. The low-rank approximation of SVD is used in this method to calculate low-rank frames. Each low-rank frame is then subjected to 2D-DWT, with the mean value of the low-frequency DWT coefficients being chosen as the hash element. Although the SVD–DWT hashing performs well against noise attacks and video rotation, interframe operations like random frame adding and dropping operations still require improvement. For high-resolution films, Tang et al. [24] proposed a stateful correlation coefficient sampling-based hash (CCSH). The CCSH uses correlation coefficient sampling for image hashing and learns from previous frames to compute hashes. It is developed for a range of video tasks, including video content-preserving operation, video authentication, and video tampering. Recently, a new hash was developed by Chen et al. [25] to trace videos on social network platforms. LL-sub band of DWT coefficients were employed in this method to reconstruct a segmentation model. The segmentation model is then decomposed using a local fluctuation descriptor based on LLE to produce a compact hash. Although the DWT–LLE hashing can withstand changes in frame rate, it is easily damaged by noise assault and video rotation.

The video hashing algorithm against common content-preserving operations and main techniques are presented in Table I, where "Yes" indicates that the algorithm is resilient against the operation and "No" indicates that the algorithm is sensitive to it. Some well-known algorithms [19], [20], [21], [22], [23] can withstand many common digital operations for the results, but these techniques are insufficient for discrimination. Due to the fact that most algorithms still find it difficult to balance robustness and discrimination, the accuracy of their video task is drastically decreased. Therefore, more effort developing high-performance algorithms with good tradeoff between robustness and discrimination in demand. Additionally, we discovered that the majority of the video hashing algorithms in use [19], [20], [21], [22], [23], [25] typically only take into consideration adding one noise model when assessing the hashing performance. More complex noise models are less discussed in video hashing research. But in actual situations, particularly in the complicated social network environment. Video sharing on social networking sites is frequently tainted by various sorts of noise, which makes social video retrieval extremely difficult.

Focus on above issue, this article we design a video hashing via secure anti-noise model (SANM) to achieve good robustness for commonly used operations and desirable discriminative capability. In the social network setting with multiple noise, high accuracy retrieval is simultaneously possible. The contributions of this work, which set distinct from other perceptual video hashing methods, are as follows.

1) We proposed an SANM with low-rank representation (LRR) and random subspace partition (RSP). Since the low-rank component of LRR is recovering the underlying subspace in noisy environment. The proposed model reconstruction with LRR is robust to noise attack. Thus, video hash derived from proposed model is beneficial to resist noise operation and ensures the high precision retrieval in the multiple noise social environment. In addition, chaotic maps are used to control the generation of RSP, which improve the security of proposed model.
2) A novel subspace decomposition description (SDD) is suggested by calculating the invariant distances of the factor matrices obtained by Tucker decomposition, and is used to decompose SANM. As the factor matrices can represent the approximate contents of the SANM and the invariant distance is insensitive to some commonly used operations and, thus, provide desirable discrimination of proposed hashing, together with preserving good robustness for some content-preserving operations.
3) Various experiments results performed on popular video Benchmark verify the effectiveness of the proposed method. Comparisons demonstrate that SANM hashing is superior to the existing well-known perceptual video hashing methods in terms of good robustness and discrimination. Social video retrieval results illustrate that the SANM hashing is better than some representative perceptual video hashing methods in terms of the P-R (Precision-Recall) curve.

The remainder of this work is organized as follows. The proposed SANM hashing is described in Section II. Experimental results toward hashing performances, e.g.,

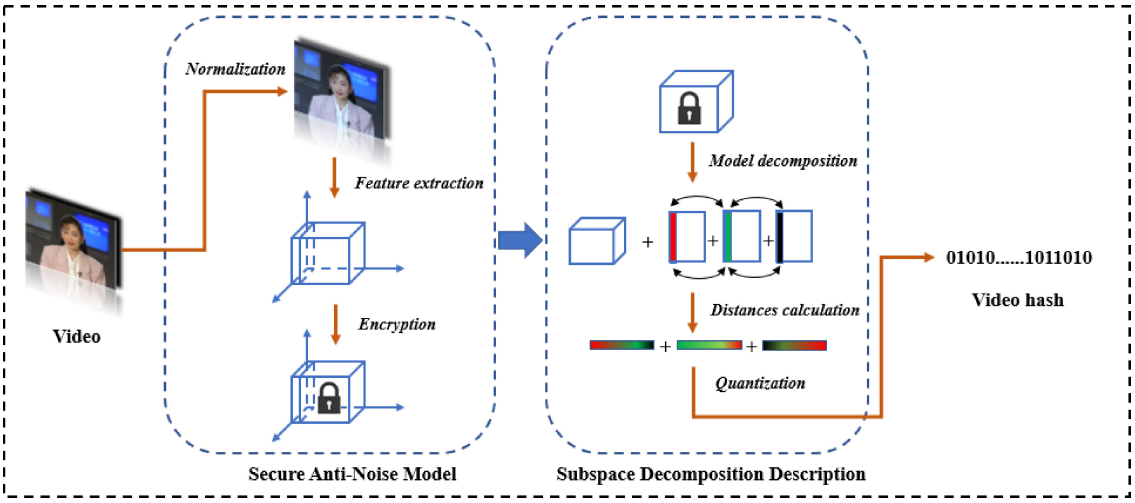

Fig. 1. Overview of SANM hashing.

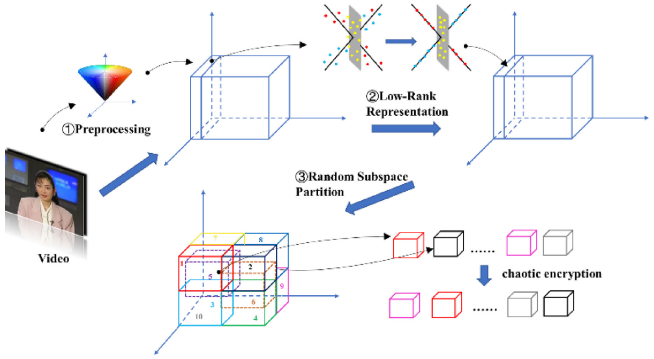

Fig. 2. Schematic of SANM.

robustness, discrimination, and key dependences and more in Section III. Conclusions are finally given in Section IV.

II. Proposed Video Hashing

Overview of SANM hashing as Fig. 1. Our work consists of four steps. A secure and anti-noise model construction will be described in Section II-A. Section II-B proposed model decomposition to a compact representation with subspace decomposition description. Section II-C specifically produces hash generation. Hash similarity scheme will be proposed in Section II-D.

A. Secure Anti-Noise Model

In this section, we propose a novel video model for robust and secure feature extraction from input video. Fig. 2 depicts the proposed model construction flow. In the first phase, color space conversion and a number of procedures are used to transform the input video into a standardized video. Subsequently LRR features from a standardized video are extracted and used to make an anti-noise video model. Finally, an SANM is ultimately produced by calculating the anti-noise model's RSP.

1) Preprocessing: Two techniques, namely, interpolation and color space conversion, are employed to create a standardized video for feature extraction in order to minimize the effects of content-preserving operations on videos. For simplicity, first define video as a series of image frames, $\mathbf{V}= [\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_N]$, where N is the frame length and $\mathbf{I}_N$ represent Nth frame of input video $\mathbf{V}$. Then, using two resampling procedures, $\mathbf{V}$ is translated to a standard video with the same resolution and number of frames. Particularly, the same-position pixels in all frames can be thought of as ordered tubes. After that, linear interpolation is used to lengthen the tube's pixels to M. The bicubic linear interpolation is then used to translate each frame of $\mathbf{V}$ into a square with the dimensions $M \times M \times 3$. Through the above calculation method, we get a standardized video model $\mathbf{V^M}= [\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_M]$.

The $\mathbf{V^M}$ is then transformed into the CIE $L^*a^*b^*$ color space, with the $\mathbf{L}^*$ component denoting the revised video. Because the $\mathbf{L}^*$ component of the CIE $L^*a^*b^*$ color space is perceptually constant and closely resembles how brightness is perceived by humans, we decided to utilize it. Let L^* denote a color pixel's brightness, and a^* and b^* denote its chromaticity coordinates. The conversion of the RGB color space to CIE L^*a^*b for each $\mathbf{V^M}$ element can be described by the following equation:

$$\mathbf{I}_i^{\mathbf{Lab}} = \left\{\mathbf{I}_i^{\mathbf{L}}, \mathbf{I}_i^{\mathbf{a}}, \mathbf{I}_i^{\mathbf{b}}\right\}, i \in [1, M] \tag{4}$$

where $\mathbf{I}_i^{\mathbf{L}}$, $\mathbf{I}_i^{\mathbf{a}}$, and $\mathbf{I}_i^{\mathbf{b}}$ are the CIE $L^*a^*b^*$ components of $\mathbf{I}_i^{\mathbf{Lab}}$, respectively, and each component is calculated by the following formulation:

$$\mathbf{I}_i^{\mathbf{L}} = 116 f\left(\frac{\mathbf{Y_i}}{Y_W}\right) - 16 \tag{5}$$

$$\mathbf{I}_i^{\mathbf{a}} = 500\left[f\left(\frac{\mathbf{X_i}}{X_W}\right) - f(\mathbf{Y_i}/Y_W)\right] \tag{6}$$

$$\mathbf{I}_i^{\mathbf{b}} = 200\left[f\left(\frac{\mathbf{Y_i}}{Y_W}\right) - f\left(\frac{\mathbf{Z_i}}{Z_W}\right)\right] \tag{7}$$

where $\mathbf{X}_i$, $\mathbf{Y}_i$, and $\mathbf{Z}_i$ make up the CIE XYZ tristimulus matrix, $X_W = 0.950456$, $Y_W = 1.0$, and $Z_W = 1{:}088754$ represent the reference white point's CIE XYZ tristimulus values, and the

formula for $f(t)$ is as follows:

$$f(t) = \begin{cases} t^{1/3}, & IF\ t > 0.008856 \\ 7.787t + \frac{16}{116}, & \text{otherwise} \end{cases}. \tag{8}$$

Moreover, the CIE XYZ tristimulus matrix $\mathbf{X}_i$, $\mathbf{Y}_i$, and $\mathbf{Z}_i$ are calculated as follows:

$$\mathbf{X}_i = 0.4125\mathbf{I}_i^{\mathbf{R}} + 0.3576\mathbf{I}_i^{\mathbf{G}} + 0.1804\mathbf{I}_i^{B} \tag{9}$$

$$\mathbf{Y}_i = 0.2127\mathbf{I}_i^{\mathbf{R}} + 0.7152\mathbf{I}_i^{\mathbf{G}} + 0.0722\mathbf{I}_i^{B} \tag{10}$$

$$\mathbf{Z}_i = 0.0193\mathbf{I}_i^{\mathbf{R}} + 0.1192\mathbf{I}_i^{\mathbf{G}} + 0.9502\mathbf{I}_i^{B} \tag{11}$$

in which $\mathbf{I}_i^{\mathbf{R}}$, $\mathbf{I}_i^{\mathbf{G}}$, and $\mathbf{I}_i^{\mathbf{B}}$ are the (red, green, and blue) components of the ith element of $\mathbf{V}^{\mathbf{S}}$. After the standard video color space conversion operation and extraction of its $\mathbf{L}^*$ component to form a new preprocessing video model $\mathbf{V}^{\mathbf{P}} = [\mathbf{I}_1{}^{\mathbf{L}}, \mathbf{I}_2{}^{\mathbf{L}}, \ldots, \mathbf{I}_M{}^{\mathbf{L}}]$.

2) Low-Rank Representation: Learning good representations from input data plays a crucial role for many social systems. In reality, the source of social video is complex, and the common situation, such as data captured by equipment, such as surveillance video is often degraded by serious complex noise [26]. LRR is an important learning technique, which decomposes input data into low-rank components and noisy parts, aiming to extract inherent low-dimensional subspaces or structures from input high-dimensional data [27]. It has been widely used in many applications [28], such as subspace segmentation, image classification, and data denoising. Define the input data $\mathbf{I}$ as an observation matrix corrupted by noise $\mathbf{E}$. Therefore, the calculation of LRR can be solved by regularization rank minimization problem [28], [29], [30]

$$\min_{\mathbf{Z},\mathbf{E}} \|\mathbf{Z}\|_* + \lambda\|\mathbf{E}\|_{2,1}, \ \text{s.t.}\ \mathbf{I} = \mathbf{I}\mathbf{Z} + \mathbf{E} \tag{12}$$

in which, low-rank matrix $\mathbf{Z}$ is the principle structures of the input data $\mathbf{I}$; $\mathbf{E}$ is a sparse matrix indicating salient components. λ is a parameter which controls the importance of the sparsity error term $\mathbf{E}$; $\|\bullet\|_*$ is the nuclear norm of a matrix, and $\|\bullet\|_{2,1}$ donates the $l_{2,1}$ norm defined as follows:

$$\|\mathbf{E}\|_{2,1} = \sum_{j=1}^{n} \sqrt{\sum_{i=1}^{n} \left([\mathbf{E}]_{ij}\right)^2}. \tag{13}$$

In practice, (12) can be converted to an equivalent optimization problem as follows:

$$\min_{\mathbf{Z},\mathbf{E},\mathbf{J}} \|\mathbf{J}\|_* + \lambda\|\mathbf{E}\|_{2,1}, \ \text{s.t.}\ \mathbf{I} = \mathbf{I}\mathbf{Z} + \mathbf{E}, \ \mathbf{Z} = \mathbf{J}. \tag{14}$$

This optimization problem can be solved by the below augmented Lagrange multiplier (ALM) problem

$$\min_{\mathbf{Z},\mathbf{E},\mathbf{J},\mathbf{Y}_1,\mathbf{Y}_2} \|\mathbf{J}\|_* + \lambda\|\mathbf{E}\|_{2,1}, + \mathrm{tr}\left[\mathbf{Y}_1^T(\mathbf{I} - \mathbf{I}\mathbf{Z} - \mathbf{E})\right] + \mathrm{tr}\left[\mathbf{Y}_2^T(\mathbf{Z} - \mathbf{J})\right] + \frac{\mu}{2}\left(\|\mathbf{I} - \mathbf{I}\mathbf{Z} - \mathbf{E}\|_F^2 + \|\mathbf{Z} - \mathbf{J}\|_F^2\right) \tag{15}$$

in which $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are the Lagrange multipliers and $\mu > 0$ donates the penalty parameter. Equation (15) can be solved by the inexact ALM method [28], [29], [30]. Fig. 3 shows the decomposition results of a sample frame. Fig. 3(a) is a sample frame. Fig. 3(b) and (c) are the low-rank component and salient component after LRR calculation in Fig. 3(a). Fig. 3(d) and (e) are the low-rank and salient components obtained by LRR calculation after adding Gaussian noise to Fig. 3(a). Fig. 3(f) is residuals between Fig. 3(e) and (c). Fig. 3(g) and (h) are the low-rank component and salient components calculated by LRR after adding mixed noise (Gaussian noise and salt and pepper noise) to Fig. 3(a). Fig. 3(i) is residuals between Fig. 3(h) and (c). It is not difficult to see that the vision of low-rank components Fig. 3(d) and (g) after adding noise are very similar to that of low-rank component Fig. 3(b). In addition, by observing Fig. 3(f) and (i), it is found that after LPP calculation, most of the noise is distributed in the salient component. In this work, low-rank matrix $\mathbf{Z}$ is considered to construct an anti-noise model for the following reasons. The effects of digital operations, such as compression, filtering, and noise on a video are treated as noise added to the video. Since LRR is robust to noise, anti-noise model reconstruction from low-rank component can improve the robustness of the proposed model. In addition, low-rank matrix $\mathbf{Z}$ is an underlying subspace in a restored noise environment, while maintaining a high visual similarity to input data $\mathbf{I}$. This means that the anti-noise model constructed by low-rank matrix $\mathbf{Z}$ is a visual approximation version of the input data $\mathbf{I}$. Therefore, the video hash generated by proposed model not only improves the robustness but also maintains the discrimination, thus, improving the balance between robustness and discrimination of the video hash. Specifically, we used LRR for each element of video model $\mathbf{V}^{\mathbf{P}}$ and extracted their low-rank matrix to form a new anti-noise video model $\mathbf{V}^{\mathbf{L}} = [\mathbf{Z_1}, \mathbf{Z_2}, \ldots, \mathbf{Z}_M]$.

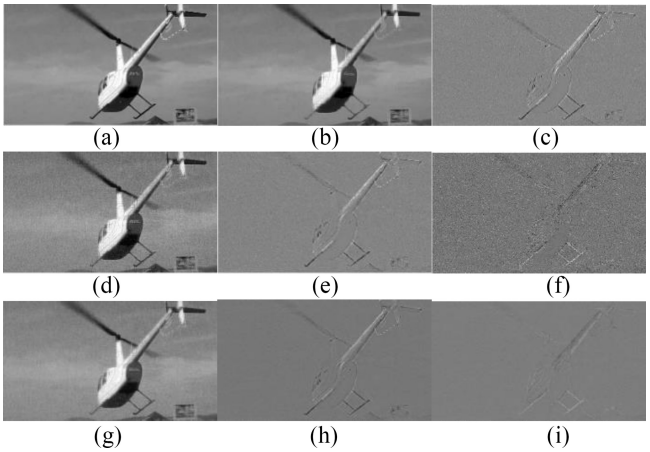

Fig. 3. LRR of a sample. (a) Is a sample frame of helicopter. (b) and (c) Are low-rank component and salient component of sample frame. (d) and (e) Are low-rank component and salient component of sample frame with noise. (f) Are the residuals of (e) and (c). (g) and (h) Are low-rank component and salient component of sample frame with mixed noise. (i) Are the residuals of (h) and (c).

3) Random Subspace Partition: To improve the security of the proposed model, we generate a subspace region consisting of some cubes for feature extraction. A random partition approach is used to generate the model in Fig. 2 in order to increase the security of the anti-noise video model $\mathbf{V}^{\mathbf{L}}$ and obtain our proposed model, SANM $\mathbf{V}^{\mathbf{S}}$. Using a secret key to control the chaotic map, the random partition will create an RSP for an $M \times M \times M$ model. Two requirements must be

met by the chosen subcubes in order for this to occur. The first requirement is that the n subcube region completely enclose the $M \times M \times M$ region. This condition ensures that all pixels within the region are used for feature calculations. The second condition is that the location and order of the subspace is chosen at random. This condition ensures that the generation of subspace has enough randomness to ensure the security of anti-noise video model. Therefore, the following strategies are adopted to calculate RSP.

First, N_1 cubes $\mathbf{S}1 = [\mathbf{V}_1{}^{\text{sub}}, \mathbf{V}_2{}^{\text{sub}}, \ldots, \mathbf{V}_{N1}{}^{\text{sub}}]$ covering all regions with subcube size $s \times s \times s$. Next, the other N_2 cubes were generated using a well-known chaotic map (skew tent map [31]). The skew tent map is defined as follows:

$$S(p) = \begin{cases} \frac{p}{q}, & p \in [0, q] \\ \frac{1-p}{1-q}, & p \in (q, 1] \end{cases} \tag{16}$$

in which $p \in [0,1]$ is an initial state of the chaotic system, and $q \in (0, 1)$ represents the control parameter. Three arrays X[N_2], Y[N_2], and Z[N_2] ($1 \le i \le N_2$), which contain the top-left coordinates of subcubes, can be used in practice to record these subcubes. Specifically, p and q are the secret keys of controlling (16) iteration to obtain $N_3 = N_2 \times 3$ elements (i.e., s_1, s_2, …, s_{N3}). The elements of N_3 are transferred to integers between [1, $M - s$] from the following equation since the N_3 elements are floating point numbers in the range [0,1]:

$$R_i = \text{mod}\Big(\text{round}\Big(s_i \times 2^{20}\Big), (M - s)\Big) + 1 \tag{17}$$

where s_i is the ith random element, mod(·) is a remainder operation, and round(·) is a rounding function. Then, through the calculation of (17), a random integer array of length N_3 is obtained, and they are assigned to the arrays X[N_2], Y[N_2], and Z[N_2] as the top-left coordinate of the position of the random subspace. Next, according to these coordinates to select N_2 cube sequences $\mathbf{S}2 = [\mathbf{V}_1{}^{\text{sub}}, \mathbf{V}_2{}^{\text{sub}}, \ldots, \mathbf{V}_{N2}{}^{\text{sub}}]$ of size $s \times s \times$s, and combine them with $\mathbf{S}1$ to get the new cube sequence $\mathbf{S}3 = [\mathbf{V}_1{}^{\text{sub}}, \mathbf{V}_2{}^{\text{sub}}, \ldots, \mathbf{V}_n^{\text{sub}}] (n = N_1 + N_2)$. To further enhance the security of the proposed model, we used another well-known chaotic map (logical map [32]) to generate a random sequence to scramble $\mathbf{S}3$. In general, the logical chaotic map is defined as follows:

$$x_{i+1} = r x_i (1 - x_i) \tag{18}$$

where r and x_0 are control parameters. Note that, the logistic map can reach a chaotic state when $r \in (3.57, 4)$ and $x_0 \in (0, 1)$. In this work, r and x_0 are the secret keys to control the (18) for iteratively getting a chaotic sequence $\mathbf{u} = [u_1, u_2, \ldots, u_n]$ with n elements. Next, the sequence u is then further mapped to the position index sequence g, where the index of the maximum value is 1 according to the relative positioning of the sequence u by the ordering of its element values. Finally, rearranging $\mathbf{S}3$ according to the index sequence $\mathbf{g}$ generates a SANM $\mathbf{V^S} = [\mathbf{V}_1{}^{\text{sub}}, \mathbf{V}_2{}^{\text{sub}}, \ldots, \mathbf{V}_n^{\text{sub}}]$, which is a randomly subspace sequence. Notably, it is almost impossible to successfully guess RPS without knowing the correct key, and key dependency is verified in Section III-D.

B. Subspace Decomposition Description

As mentioned above, extracting content features of video using an SANM can indeed greatly improve robustness. By calculating the SANM of the input video, a safe random subspace sequence $\mathbf{V^S}$ is obtained. For a compact video hash, we need to further reduce the dimension of $\mathbf{V^S}$. Thus, we design a new subspace decomposition descriptor (SDD) to decompose the proposed model to generate content-based and compact representation. Specifically, $\mathbf{V^S}$ is regarded as a third-order tensor sequence. Tensor decomposition is used to obtain the low-dimensional representation of $\mathbf{V^S}$, and its invariant distances are extracted to form the SDD of $\mathbf{V^S}$.

1) Tensor Decomposition: Tensor decomposition [33] is a useful data processing technology, which has been successfully applied in many disciplines [34], [35], [36], such as data analysis, graph representation, computer vision, signal processing, etc. In this work, a well-known algorithm called Tucker decomposition [37] is used to implement tensor decomposition to derive SDD. There are many research applications based on Tucker decomposition in knowledge graph completion, data compression, density functional calculations, and noise reduction [38], [39]. For an input tensor $\mathbf{T} \in^{P \times Q \times S}$ (a third-order tensor), Tucker decomposition will decompose it into a core tensor $\mathbf{G} \in^{I \times J \times K}$ and three factor matrices $\mathbf{A} \in^{P \times I}$, $\mathbf{B} \in^{Q \times J}$, and $\mathbf{C} \in^{S \times K}$. In general, $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are thought of as the primary components in each mode, reflecting the fundamental structure of $\mathbf{T}$ [40], while $\mathbf{G}$ generally denotes the level of interaction between the different components. Specifically, $\mathbf{G}$ is viewed as a compressed version of $\mathbf{T}$ when the conditions are satisfied that I, J, and K is less than P, Q, and S. In general, the third-order Tucker decomposition can be expressed by the mathematical formula

$$\mathbf{T} \approx [\![\mathbf{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} g_{i,j,k} \big(a_i \circ b_j \circ c_k\big) \tag{19}$$

where, ai is the column vector of $\mathbf{A}$, bj, and c_k are the column vectors of $\mathbf{B}$ and $\mathbf{C}$, $g_{i,j,k}$ donate the element of $\mathbf{G}$ and the symbol "∘" implis outer product between two vectors. The Tucker decomposition in (20), calculated by elements, is

$$t_{p,q,s} \approx \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} g_{i,j,k} a_{p,i} b_{q,j} c_{s,k}, \quad p = 1, \ldots, P \\ q = 1, \ldots, Q, s = 1, \ldots, S \tag{20}$$

in which, $t_{p,q,s}$ is the element of $\mathbf{T}$, so $a_{p,i}$, $b_{q,j}$, and $c_{s,k}$ are the element of $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$. In practice, (19) can be converted to an equivalent optimization problem as follows:

$$[\![\mathbf{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = \arg \min_{g_{i,j,k}, a_i b_j c_k} \left\| \mathbf{T} - \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} g_{i,j,k} \big(a_i \circ b_j \circ c_k\big) \right\|_2 \tag{21}$$

in which $\| \|_2$ is the Frobenius norm. Equation (21) can be solve by higher-order orthogonal iteration (HOOI) method, and more details of this algorithm can be referred to [41].

2) Invariant Distances: Since content-preserving operations typically only slightly change a point's position, they have little effect on vector distances and are, therefore, invariant to some common digital operations. This finding can be used to create an effective compression method [42] that uses vector distance as the hashing element. Suppose $\mathbf{U}$ as a matrix of n vectors

$$\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n]. \tag{22}$$

Let $\mathbf{u}_i = [u_i(1), u_i(2), \ldots, u_i(m)]$ be the ith row of $\mathbf{U}$ and then convert it to $\mathbf{z}_i$ using the following rule:

$$z_i(j) = \left| \frac{u_i(j) - m_i}{w_i} \right| \tag{23}$$

where w_i and m_i are the standard deviation and mean value of $\mathbf{u}_i$, $u_i(j)$, and $z_i(j)$ are the jth elements of $\mathbf{u}_i$ and $\mathbf{z}_i$, respectively. The absolute value operation is represented by the symbol "$|\bullet|$." The result of the aforementioned process is a normalized matrix $\mathbf{U}$. The reference vector $\mathbf{r} = [r(1), r(2), \ldots, r(n)]$ is then calculated, where r is defined as

$$r(j) = \frac{1}{n} \sum_{i=1}^{n} u_i(j). \tag{24}$$

The Euclidean distance $d(i)$ between $\mathbf{z}_i$ and $\mathbf{r}$ is computed by the equation

$$d(i) = \sqrt{\sum_{j=1}^{m} (z_i(j) - r(j))^2}. \tag{25}$$

It is worth that $d(i)$ is a decimal that requires many bits to store (e.g., 32 bits for floating-point numbers and 64 bits for doublets). Therefore, in order to reduce storage space, $d(i)$ is quantized as an integer $f(i)$ by the following equation:

$$f(i) = \text{round}[d(i) \times 100 + 0.5]. \tag{26}$$

Finally, After the above method of operation, we get a integer vectors $\mathbf{f} = [f(1), f(2), \ldots, f(n)]$.

In this work, we develop the SDD based on the considerations below using the factor matrices from Tucker decomposition and invariant distances. The orthogonal factor matrices of the Tucker decomposition can be thought of as a low-dimensional content representation of the input tensor since they can reflect the intrinsic structure of the original tensor. This advantage ensures the discrimination of SDD. In addition, these perceptual factor matrices' Euclidean distances between vectors are invariant to frequently used digital manipulations to videos, which benefits in the compactness and robustness of SDD. Therefore, using SDD to decompose $\mathbf{V^S}$ to construct a compact video hash method can ensure the discriminant ability of the proposed algorithm and robustness to some common digital operations. Based on the above analysis, we designed the calculation method of SDD as follows.

For each element of $\mathbf{V^S}$, Tucker decomposition method is adopted to decompose each element of $\mathbf{V^S}$, and their factor matrix is extracted to form three corresponding matrix arrays $\mathbf{A}(i)$, $\mathbf{B}(i)$, and $\mathbf{C}(i)(1 \le i \le n)$, where n corresponds to the number of $\mathbf{V^S}$. The mean of each row is computed for the factor matrix $\mathbf{A}(i)$, and the resulting feature vector is as follows:

$$\mathbf{e}^{\mathbf{A}(i)} = \left[e_1^{\mathbf{A(i)}}, e_2^{\mathbf{A(i)}}, \ldots, e_P^{\mathbf{A(i)}} \right]^T \tag{27}$$

where $e_j^{\mathbf{A}(i)}$ is the mean value for the jth row of $\mathbf{A}(i)$. Next, a factor matrix $\mathbf{A}(i)$-based feature matrix $\mathbf{E^A}$ is available as follows:

$$\mathbf{E^A} = \left[\mathbf{e}^{\mathbf{A}(1)} \mathbf{e}^{\mathbf{A}(2)}, \ldots, \mathbf{e}^{\mathbf{A}(n)} \right]. \tag{28}$$

Similarly, for the factor matrix $\mathbf{B}(i)$, we calculate the mean of each row and get the following eigenvectors:

$$\mathbf{e}^{\mathbf{B(i)}} = \left[e_1^{\mathbf{B(i)}}, e_2^{\mathbf{B(i)}}, \ldots, e_Q^{\mathbf{B(i)}} \right]^T \tag{29}$$

where $e_j^{\mathbf{B}(i)}$ is the mean of the jth row of $\mathbf{B}(i)$. Next, a factor matrix $\mathbf{B}(i)$-based feature matrix $\mathbf{E^B}$ is available as follows:

$$\mathbf{E^B} = \left[\mathbf{e}^{\mathbf{B}(1)}\ \mathbf{e}^{\mathbf{B}(2)}, \ldots, \mathbf{e}^{\mathbf{B}(n)} \right]. \tag{30}$$

Then, we calculate the factor matrix $\mathbf{C}(i)$ in the same way and get the vector

$$\mathbf{e}^{\mathbf{C(i)}} = \left[e_1^{\mathbf{C(i)}}, e_2^{\mathbf{C(i)}}, \ldots, e_S^{\mathbf{C(i)}} \right]^T \tag{31}$$

where $e_j^{\mathbf{C}(i)}$ is the mean of the jth row of $\mathbf{C}(i)$. The feature matrix $\mathbf{E^C}$ based on factor matrix $\mathbf{B}(i)$ is shown as follows:

$$\mathbf{E^C} = \left[\mathbf{e}^{\mathbf{C}(1)}\ \mathbf{e}^{\mathbf{C}(2)}, \ldots, \mathbf{e}^{\mathbf{C}(n)} \right]. \tag{32}$$

For the feature matrices $\mathbf{E^A}$, $\mathbf{E^B}$, and $\mathbf{E^C}$, their invariant distances were calculated, respectively, by referring to (22)–(26) to obtain three SDDs $\mathbf{f^A}$, $\mathbf{f^B}$, and $\mathbf{f^C}$. Finally, we concatenate $\mathbf{f^A}$, $\mathbf{f^B}$, and $\mathbf{f^C}$ to get the final hash

$$\mathbf{h} = \left[f^{\mathbf{A}}(1), \ldots, f^{\mathbf{A}}(n), f^{\mathbf{B}}(1), \ldots, f^{\mathbf{B}}(n), f^{\mathbf{C}}(1), \ldots, f^{\mathbf{C}}(n), \right]. \tag{33}$$

Therefore, our hash length is $H = n \times 3$ integer.

C. Hash Generation

See Algorithm 1.

D. Similarity Evaluation

Between two video hashes, there are various measure similarity functions [43]. The distance between two similar videos should be smaller and the distance between two different videos should be larger for the similarity measurement of video hashes $\mathbf{h}_1$ and $\mathbf{h}_2$. In particular, SANM hashing is an integer representation, and the L2 norm is exploited to gauge how similar our hashes are to one another. It can be stated as follows:

$$d_h(\mathbf{h}_1, \mathbf{h}_2) = \sqrt{\sum_{i=1}^{H} [h_1(i) - h_2(i)]^2}. \tag{34}$$

If $\mathbf{h}_1$ and $\mathbf{h}_2$ are two distinct hashes produced by hash algorithms, and $h_1(i)$ and $h_2(i)$, respectively, represent the ith element of each hash. To establish how similar two videos

Algorithm 1 Proposed SANM Hashing

Input: An input Video $\mathbf{V}$;
M: the size of standardized video model;
p, q, r, x_0: the secret keys of the chaotic map;
λ: the control parameter the of the sparsity error term
s: the size of sub-cube;
n: the number of sub-cubes;
I, J, K: the selected value of tucker decomposition.
Output: the hash sequence $\mathbf{h} = [h_1, h_2, \ldots, h_H]$;
1: Input an input video $\mathbf{V}$ and calculate its secure anti-noise model (SANM).
I. Conduct interpolation and color space transformation operations on the input video $\mathbf{V}$ to obtain a standardized preprocessing video model $\mathbf{V^P} \in R^{M\times M\times M}$;
II. The anti-noise video model $\mathbf{V^L}$ is obtained by extracting LRR from each element of $\mathbf{V^P}$ under the control of parameter λ. Calculation of LRR can be solved by regularization rank minimization problem (12), and solve this optimization by the inexact ALM method.
III. Using keys p, q, r and x_0 to control chaotic mapping, calculate a RSP with n subcube size $s\times s\times s$ of $\mathbf{V^L}$ to get secure anti-noise model $\mathbf{V^S}$.
2: Factor matrix arrays $\mathbf{A}(i)$, $\mathbf{B}(i)$ and $\mathbf{C}(i)$ generating from $\mathbf{V^S}$ decomposition with tucker decomposition (TD) under the selected values I, J and K. TD through the minimization problem (22), and solve this optimization with a well-known method (HOOI).
3: Subspace decomposition descriptors (SDD) are generated by extracting compact and robust low-dimensional representations from $\mathbf{A}(i)$, $\mathbf{B}(i)$ and $\mathbf{C}(i)$ using the invariant vector distances.
I. Construct normalized matrices $\mathbf{E^A}$, $\mathbf{E^B}$ and $\mathbf{E^C}$ derived from $\mathbf{A}(i)$, $\mathbf{B}(i)$ and $\mathbf{C}(i)$;
II. Calculate the reference vector $\mathbf{r}$ of the normalized matrix;
III. By calculating the Euclidean distance between $\mathbf{E^A}$, $\mathbf{E^B}$ and $\mathbf{E^C}$ and $\mathbf{r}$, three invariant distances are obtained and further converted into integer subspace decomposition descriptors $\mathbf{f^A}$, $\mathbf{f^B}$ and $\mathbf{f^C}$;
4: Proposed hashing $\mathbf{h}$ derive from subspace decomposition descriptors, by concatenation $\mathbf{f^A}$, $\mathbf{f^B}$ and $\mathbf{f^C}$.

are, a predetermined threshold T is typically employed. They can be distinguished as visually similar videos when there is $d_h<T$. In other words, a matching video can be recognized as a similar video if the L2 norm of two hashes is less than a predetermined threshold. If not, they can be assumed to be the visually distinct versions.

III. Experimental Results

In this section, the parameters setting of SANM hashing is described as follows. The standardized video size is $128 \times 128 \times 128$. The control parameter the of the sparsity error term is 0.85. The subcube size is $32 \times 32 \times 32$; The number of subcubes is 128. The selected values of the of Tucker decomposition are all 1. In other words, the parameter input of the SANM hash algorithm are: $M = 128$, $\lambda = 0.85$, $s = 32$, $N_1 = 64$, $N_2 = 64$, $n = 128$, $I = 1$, $J = 1$, and $K = 1$. The length of SANM hashing is, therefore, 384 integers. The video database and digital operations for the robustness configuration are explained in Section III-A of the following sections. Results of the experiments on robustness and discrimination are shown in Sections III-B and III-C. Sections III-D and III-E conduct research on key dependence and a few important parameters that affect hash performance. Sections III-F and III-G then examine performance comparisons and video social retrieval.

A. Experimental Setting and Database

Comprehensive tests are conducted using two well-known public data sets: 1) Amsterdam library of ordinary videos (ALOV300++) [45] and 2) visual object tracking benchmark 2015 (VOT2015) [44]. Fig. 5 displays thumbnails of some sample videos. Fig. 5(a) and (b) shows sample videos from ALOV300++ and thumbnails of benchmark videos from VOT2015, respectively. The following is a comprehensive description of the used data sets.

VOT2015: This data set includes 60 brief sequences (e.g., a car, a crosswalk, and a piece of nature) spread over numerous scenarios. The frame resolution is between 320×240 and 960×540, and the frame length is between 140 and 1500. The well-known programs, such as Photoshop, MATLAB, and STIRMARK, are exploited to carry out robustness operations to create content-preserving videos. The three types of content-based attacks were modified (e.g., intraframe, interframe, and geometric attacks). Examples include video rotation, brightness adjustment, noise attack, mpeg compression, frame scaling, and random frame dropping. In specifically, three typical single noise attack models as well as their combination attack models and mixed attacks are also introduced to the robustness evaluation in order to validate the performance of the algorithm against multiple noise attacks in the social network environment. The 12 content preservation operations and their parameter settings are described in detail in Table II and some visualization of different multiple types of noise models are listed in Fig. 4. There are 110 different attack videos for every original video, which means that each original video has 110 similar videos. This results in a total of 6600 visually similar video pairs. Thus, a total of 6660 videos were assessed for robustness, including 60 original videos and 6600 visually comparable videos.

ALOV300++: This data set provides 14 types of video clips, such as light, transparency, shape and clutter. Each category contains more than ten video clips, a total of 314 different videos. In addition, these videos contain a variety of video resolutions. The minimum resolution is 192×144, and the high resolution reaches 720p and 1080p. The videos of the ALOV300++ are used for discrimination analysis.

B. Performance of Robustness

The videos from VOT2015 and its content-preserving videos, which were introduced in the previous section, are used in this section to assess perceptual robustness. They are substantial enough to show the validity of the experimental findings. To be more precise, we use the actual videos from VOT2015 and its visually similar videos to compute their L2 norms similarity. Fig. 6 displays the mean L2 norm for the

TABLE II
CONTENT-PRESERVING OPERATIONS AND THEIR PARAMETER SETTING

Operation	Parameter	Parameter value	Number
Brightness adjustment	Photoshop's scale	-20,-15,…,15,20	8
White gaussian noise	Variance	0.01,0.02,…,0.09,0.1	10
Salt and pepper noise	Density	0.01,0.02,…,0.09,0.1	10
Speckle noise	Variance	0.01,0.02,…,0.09,0.1	10
White gaussian & Salt and pepper noises	Hybrid parameter	[0.01,0.01],…,[0.1,0.1]	10
Salt and pepper & speckle noises	Hybrid parameter	[0.01,0.01],…,[0.1,0.1]	10
Horizontal distribution of noise combinations	Combination parameter	[0.01,0.01,0.01],…,[0.1,0.1,0.1]	10
Vertical distribution of noise combinations	Combination parameter	[0.01,0.01,0.01],…,[0.1,0.1,0.1]	10
MPEG-2 compression	Kilobit per second	100,200,…,1000	10
Frame scaling	Ratio	0.8,0.85,…,1.2	8
Random frame dropping	Frame number	1,2,5,10,15,20	6
Video rotation	Angle(degree)	-2,-1.25,-0.75,0.75,1.25,2	6
Total			110

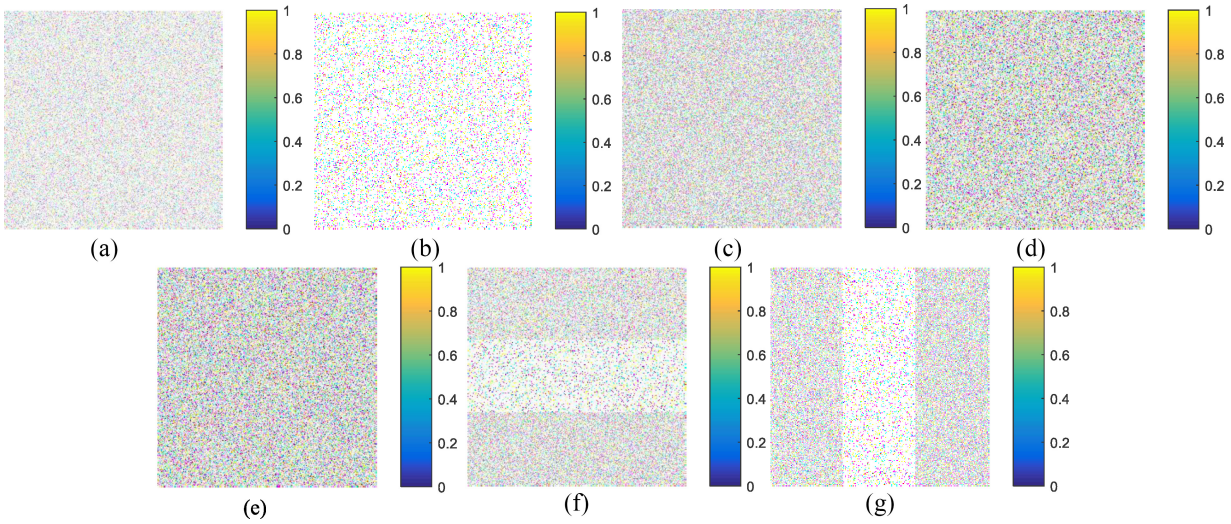

Fig. 4. Visualization of different multiple types of noises. (a) Is a Gaussian noise model, (b) is a salt and pepper noise model, (c) is a speckle noise model, (d) is a mixture model of Gaussian noise and salt and pepper noise, (e) is a mixture model of speckle noise and salt and speckle noise, (f) is a horizontal distribution model of three kinds of noise, and (g) is a vertical distribution model of three kinds of noise.

digital operation parameters. The horizontal axis represents the calculation parameters taken from Table II, while the vertical axis represents the L2 norm. Fig. 6(a)–(l), respectively, represents 12 different operations, such as brightness adjustment, White Gaussian noise, Mpeg2 compression, and random frame dropping. In order to facilitate observation of a reference threshold, a red horizontal line is drawn in the figure. Note that a smaller L2 norm between the original video and its visually similar version implies greater robustness, refer to (1) and its definition. It is clear from Fig. 6 that all mean L2 norm values are below the reference threshold. Additionally, it is not difficult to discover that brightness adjustment, White Gaussian noise, Salt and pepper noise, Speckle noise, White Gaussian & salt and pepper noise, Salt and pepper & speckle noises, Horizontal distribution of noise combinations, vertical distribution of noise combinations, MPEG-2 compression, Frame scaling, and Random frame dropping are particularly excellent in operation. Their mean values of L2 norm is all smaller than the reference threshold, which means that our algorithm has strong robustness for these 11 commonly used operations when the threshold T is set to 200. In particular, it can be seen from Fig. 6(b)–(h) that the proposed algorithm robust against multiple noise attacks, which is attributed to the use of SANM to extract video features. A black color block fills the video rotation after it has been turned. More content is filled as the video's rotation angle rises. In other words, both rotation and a fill-in operation were applied to the video rotation procedure. Because of this, L2 norm of video rotation are greater than those of other single operations. Therefore, a very tiny number of videos will be mistakenly classified as different videos if $T = 200$ is chosen as the threshold. False categorization will not happen when T is 220.

C. Performance of Discrimination

Different hash codes must be generated from videos with various visual contents in order to discrimination. This means that each pair of different videos' L2 norms should be as large as possible. To this end, discrimination tests are carried out using ALOV300++. The ALOV300++ data set is described in the section before this one. In Fig. 5(b), few examples are displayed. It is important to note that while ALOV300++ does contain many videos with diverse visual material, some of them are nearly identical videos with similar content that were shot in the same scene. These almost identical videos may look

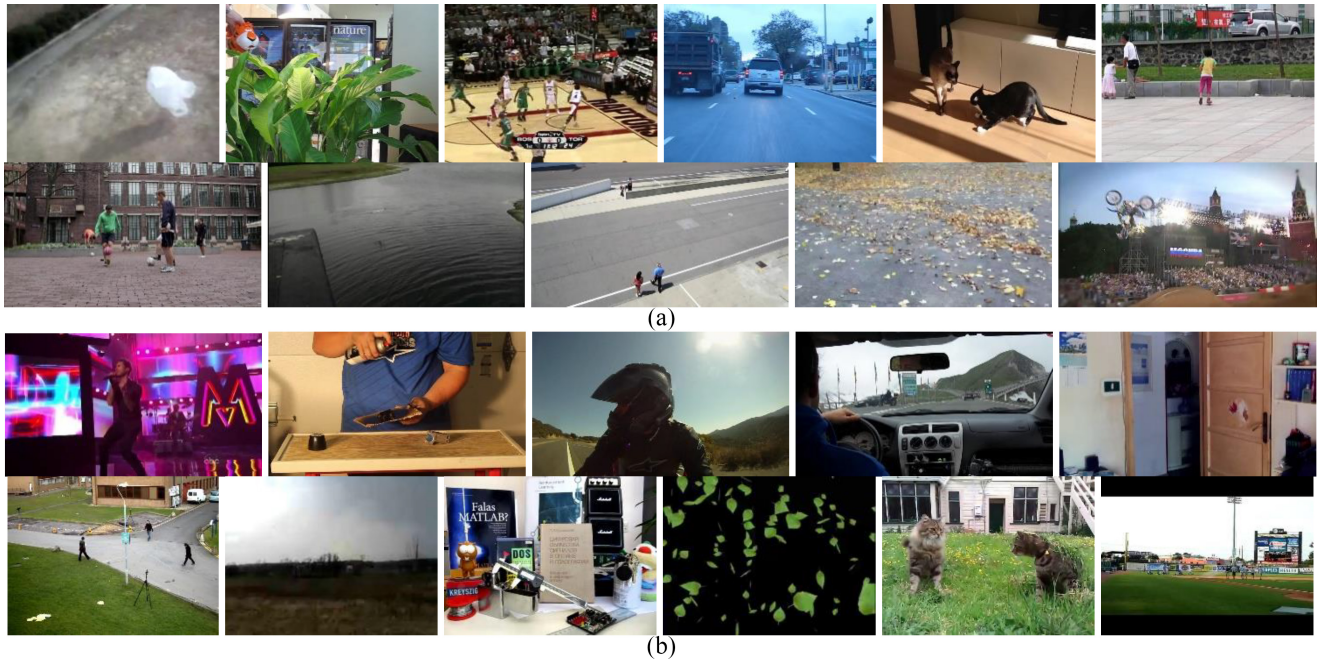

Fig. 5. Thumbnails of some typical videos of open databases. (a) Visual object tracking benchmark. (b) ALOV300++s.

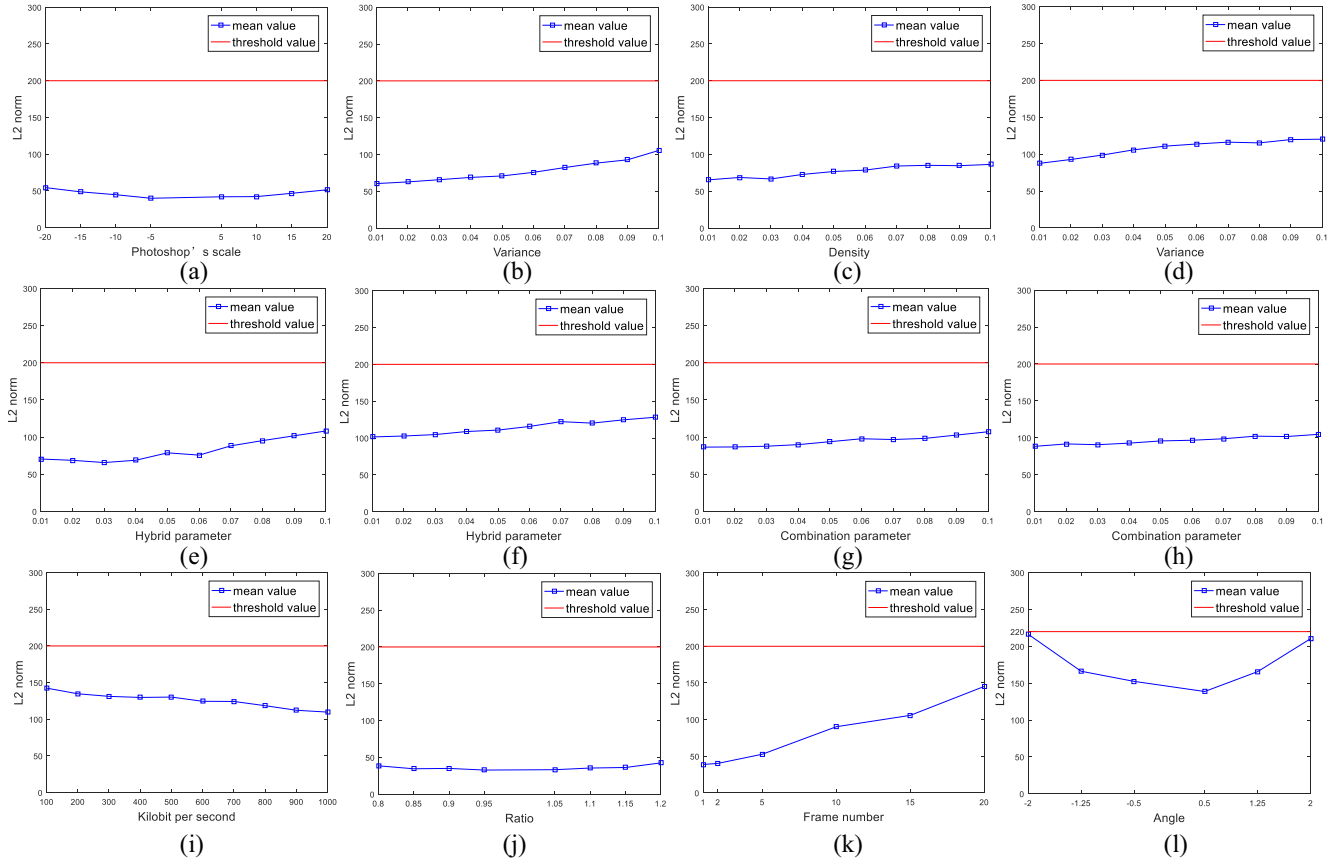

Fig. 6. Robustness results under different operations. (a) Brightness adjustment. (b)White Gaussian noise. (c) Salt and pepper noise. (d) Speckle noise. (e) White Gaussian and salt and pepper noise. (f) Salt and pepper and speckle noises. (g) Horizontal distribution of noise combinations. (h) Vertical distribution of noise combinations. (i) MPEG-2 compression. (j) Frame scaling. (k) Random frame dropping. (l) Video rotation.

more alike visually than certain versions with intact content, making it more challenging to distinguish between them. As a result, the experimental findings from ALOV300++ will be stronger. In particular, we produced hashes for ALOV300++ videos and evaluated the L2 norm between the original hash and their other hashes. The distribution of these values is

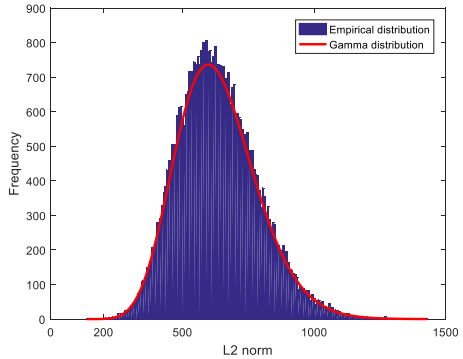

Fig. 7. Hash distance distribution of different videos.

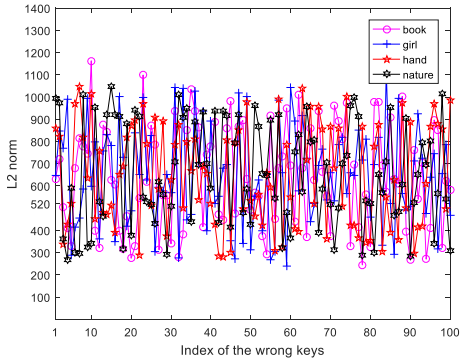

Fig. 8. L2 norms between hashes of the test video generated by different keys.

TABLE III
CHI-SQUARE RESULT FOR L2 NORM

Distribution	Estimated parameters	$\chi 2$
Normal	μ=635, σ=152	22648
Lognormal	μ=6.42, σ=0.24	6.39×10^5
Rayleigh	β=461.57	2.93×10^4
Weibull	β=4.35, η=694.65	7.36×10^6
Gamma	a=17.28, b=36.71	2827

shown in Fig. 7, where the vertical axis is the frequency, and the horizontal axis is the L2 norm. We discovered that the majority of different videos had distances that are bigger than the T value of 200, indicating that our SANM hashing can discriminate between different content videos effectively. It is obvious that a good threshold can effectively separate different videos from content-preserving ones. The L2 norm of the overlap interval between content-preserving videos and different videos is suggested to determine the threshold. Note that, the empirical distribution of L2 norm by calculation from ALOV300++ is shown in Fig. 7, which is used to the theoretical analysis. The Chi-square statistical test [46] was used to evaluate the actual distribution, identify the nearest distribution, and calculate the likelihood of a collision. The results of the Chi-square test for a variety of distributions, including the normal distribution, lognormal distribution, Rayleigh distribution, Weibull distribution, and Gamma distribution, are shown in Table III. These distances are first quantified with a step size of 5, and then a set of discrete points is obtained, since the L2 norm is a float point number. Next, the parameters of these distributions are estimated by maximum-likelihood estimation, and $\chi 2$ was calculated as follows:

$$\chi^2 = \sum_{i=0}^{L} \frac{(x_i - x_{\text{num}} y_i)^2}{x_{\text{num}} y_i} \tag{35}$$

where x_i is the frequency with which the L2 norm occurs when trial number x_{num} is i. L represents the total number of discrete points, and y_i represents the probability at position i as determined by the probability density function. The Gamma distribution, which is the real distribution represented by the red curve in Fig. 7, obviously has the smallest $\chi 2$. We can assume that our hash distance follows the Gamma distributions with $a = 17.28$ and $b = 36.71$ because the $\chi 2$ value of the Gamma distribution is minimum. As a result, we use the formula below to determine the collision probability corresponding to the hash by a predetermined threshold T

$$P(d_H \le T) \int_0^T \frac{1}{b^a \Gamma(a)} x^{a-1} e^{\frac{-x}{b}} dx \tag{36}$$

in which, $\Gamma(a)$ is a gamma function as follows:

$$\Gamma(a) = \int_0^{+\infty} t^{a-1} e^{-t} \text{dt}. \tag{37}$$

D. Performance of Key Dependence

In Section II-A, we develop an RPP strategy in association with a chaotic mapping encryption technique to increase the anti-noise video model's security. In particular, the control parameters r and x_0 of the logistic map's control parameters as well as the control parameters p and q of the tent map are employed as encryption keys. For SANM hashing, key dependence requires that the L2 norm between hash sequences of a video controlled by different keys be sufficient larger. In this work, key dependent performance is verified using standard benchmark videos from the VOT2015. Specifically, for a video, with all other parameter constant, first generate a hash sequence from a correct set of keys and a set of 100 wrong keys, respectively, and then calculate the L2 norm between them. For space limitation, results of these video book, girl, hand, and nature, i.e., are taken for example. Therefore, the distance result is shown in Fig. 8, where the x-axis is the index of the wrong keys and the y-axis is the L2 norm. It is observed that the minimum value of L2 norm is 245, and most of the distance is within this range [300 1200]. Note that, the above results are much higher than the reference T of 200. They are almost spread across the L2 norm of different videos, which verifies that different keys lead to different video hashes. This result also shows that the SANM hashing is key dependence.

E. Effect of Key Parameter on Hash Performances

In this section, we analyze the impact of the crucial SANM hashing parameters M, s, λ, I, J, and K on hashing performance using the ROC curve. Because the ROC curve is independent of particular thresholds, it serves as a comprehensive indicator of robustness and discrimination. It is obvious that the

ROC curve in the top left achieves a decent tradeoff between robustness and discrimination since each point on the curve is produced by a pair of P_{FPR} and P_{TPR}.

True Positive Rate: P_{TPR} indicates that the content is correctly identified, which equivalent to robustness. It can be determined by

$$P_{\text{TPR}}(d_H \le T) = \frac{N_{\text{same}}}{N_{\text{identical}}} \tag{38}$$

in which, $N_{\text{identical}}$ presents total pair of similar videos and N_{same} presents pair of similar videos identified as similar videos. Generally speaking, robustness increases with P_{TPR} size.

False Positive Rate: Indicates that the different videos are wrongly accepted as content-preserving videos. It can be determined by

$$P_{\text{FPR}}(d_H \le T) = \frac{N_{\text{similar}}}{N_{\text{different}}} \tag{39}$$

where $N_{\text{different}}$ stands for the total number of different pairs of videos and N_{similar} for the number of distinct pairs of videos that have been deemed similar. Similarity, the smaller the P_{FPR}, the better the discrimination. Two additional frequently used indicators [receiver operating characteristic curve (ROC) [47] and area under the ROC curve (AUC)] are used to assess the criteria for more direct observation and quantification of experimental data.

ROC: ROC curve is obtained by a set of points (P_{FPR}, P_{TPR}), where each point corresponds to a T. Be aware that a good tradeoff between robustness and discrimination is suggested by the ROC curve's location toward the upper left.

AUC: The AUC's numerical statistics are represented by AUC. The tradeoff performance is often better the larger the AUC.

In this section, the parameters (M, s), λ, and (I, J, K) will be divided into three groups for discussion, where M is the standardized video size, s indicates the size of subcube, λ is the control parameter the of the sparsity error term, and I, J, K is effect of the parameters of Tucker decomposition. During the experiment, the VOT2015 and ALOV300++ were again used to change only one parameter (M and s), λ, and (I, J, K) while keeping other parameters unchanged.

For the first part, several sets of changing parameters M and s are used to discussed on the performance of SANM hashing. These two parameters are discussed together because they affect both the length and generation time of the hash. In this experiments, four parameters are tested: 1) ($M = 128$, $s = 32$); 2) ($M = 128$, $s = 64$); 3) ($M = 256$, $s = 32$); and 4) ($M = 256$, $s = 64$). $N_1 = M/s$, for comparison we set the random subblocks N_2 and N_1 to be the same, thus, the hash length corresponding to the above parameter combinations are 384, 96, 3072, and 384. For each parameter values, robustness, and discrimination validation test and the ROC curves are formed by a series of points (P_{FPR}, P_{TPR}) and presented in Fig. 9. It is clear that the ROC curves of different (M, s) values are significantly different. This means that the (M, s) selection will have important influence on hashing performance. Specifically, we also calculate the average time for each video

TABLE IV
AUCs AND COMPUTATIONAL TIME COMPARISONS AMONG DIFFERENT M *and* S VALUES

M	s	AUCs	Hash Length	Time(s)
128	32	0.9981	384 integers	32.25
128	64	0.9854	96 integers	22.36
256	32	0.9975	3072 integers	48.78
256	64	0.9958	384 integers	36.28

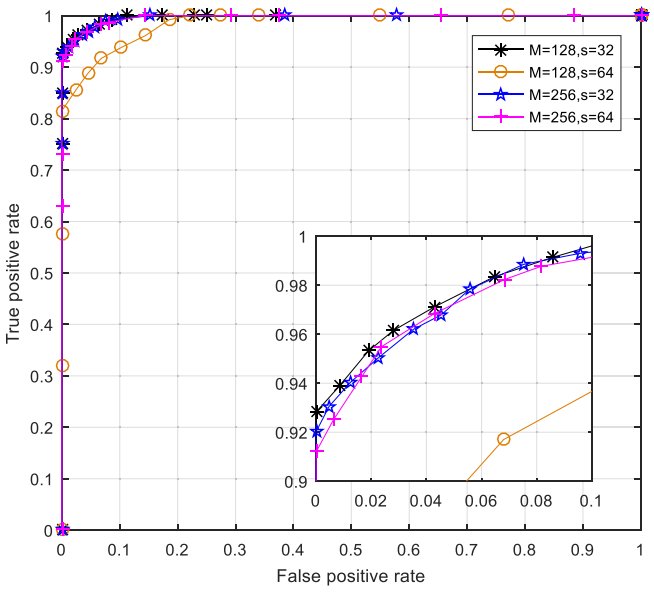

Fig. 9. ROC curves under different parameters.

to generate a hash with the personal laptops (2.26 GHz Intel Core i7-8750H CPU, 16.0 GB RAM). According to the analysis of experimental result Table IV, ($M = 128$, $s = 32$) has a better balance of robustness and discrimination than other parameters, and its hash length and computation time are moderate.

For the second part, the control parameter the of the sparsity error term λ will be discussed, and the used value are 0.75, 0.85, 0.95. According to the analysis in Section II-A, we found that most of the noise is concentrated on the sparse matrix. Appropriate λ value can well determine the weight allocation of sparse matrix and low-rank component in noisy environment, ensure the approximation of low-rank component and input content to the maximum extent, and overcome the impact of noise attack. Fig. 10 shows the ROC curve comparisons between various values. It is obvious that when λ is 0.85, the tradeoff between robustness and discrimination is ideal. This is because too small or too large a value of λ leads to limited or redundant content of low-rank components, affecting robustness and discrimination performance. According to the analysis of experimental data, the most appropriate value of λ is 0.85 in this experiment.

The third section discusses the impact of the I, J, and K Tucker decomposition parameters on hash performance. To get a decent dimension reduction effect, the parameter values for I, J, and K should typically be smaller than the order of the tensor. There is a total of eight combinations using the chosen values for I, J, and K, respectively. Fig. 11 depicts the

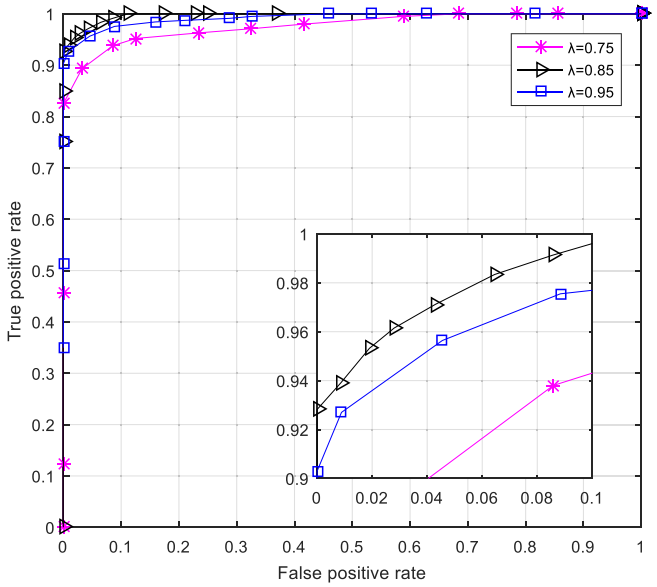

Fig. 10. ROC curves of different λ values.

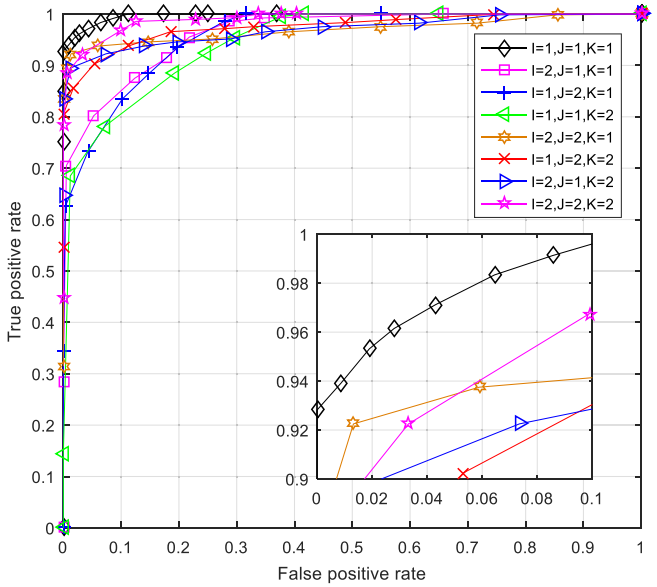

Fig. 11. ROC curves of different I, J, and K values.

ROC curve comparison between the values of various I, J, and K combinations for an obvious comparison. When $I = 1$, $J = 1$, and $K = 1$, it is noticed that the ROC curve is closer to the top-left corner than it is for other parameters. This indicates that when $I = 1$, $J = 1$, and $K = 1$, the tradeoff between robustness and discrimination for SANM hashing is advanced.

The impact of various noise intensities on algorithm performance is examined in the fourth section. Different noise intensities will be tested for single noise (white Gaussian noise) and mix noise (white Gaussian noise and salt and pepper noise). Algorithm AUC values were examined and tallied in order to measure experimental findings under various noise levels. The results show that the algorithm's performance starts to rapidly deteriorate in the presence of single noise when the

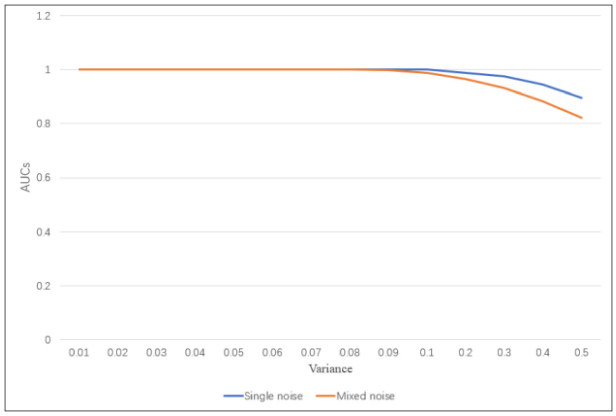

Fig. 12. Effectiveness of the algorithm on different noise intensity.

variance approaches 0.3. When there is mixed noise and the variance is more than 0.1, the algorithm's performance gradually degrades. As a result, mixed noise affects the algorithm more than single noise, and our algorithm can continue to operate effectively under various model noises with a variance of 0.1. This is so that the hash algorithm can have strong noise resistance due to the SANM presented in this research. For more details on the performance of this method at different noise intensities presented in Fig. 12.

F. Performance Comparisons

In this part, we compare the advantages of the proposed hashing with a few state-of-the-art methods. The two main characteristics of video hashing are, generally: 1) robustness and 2) discrimination. For the perceptive hash algorithm, its robustness and discriminability can be independently controlled by a specific threshold. How to strike a good balance between the two remains a challenging task. To verify the advantages of the proposed hashing methods, several well-known hashing methods, including DCT-DST hashing method [19], DWT-IM hashing methods [22], SVD-DWT hashing method [23], and CCSH hashing method [24] are compared. The compared methods have been recently published in reputable international journals, some of which are based on spatial feature methods, such as SVD-DWT hashing and CCSH and based on spatial and temporal features methods, such as DCT-DST hashing and DWT-IM hashing. All input videos are converted to 256 × 256 × 256 before the hash is generated, and the parameters and hash similarity measures are configured for these comparison methods using the settings they report. The videos used in the Sections III-B and III-C are taken to validate performances of the evaluated video hashing algorithms.

Fig. 13 illustrates the curves for the different methodologies. Zoom in on the upper left to lower right corners of these curves to more clearly compare the local details. Compared to the other approaches, the SANM hashing method's curve is more inclined to the upper left corner. As a result, the hash method suggested in this article performs better in terms of classification than other methods. The AUC values of the proposed hashing method and the comparison method are

TABLE V
TPR COMPARISONS AMONG DIFFERENT ALGORITHMS UNDER DIFFERENT OPERATIONS WHEN FPR CLOSE TO 0

Operation	DCT-DST Hashing	DWT-IM hashing	SVD-DWT hashing	CCSH hashing	SANM Hashing
Brightness adjustment	0.9233	0.9917	1.0	0.8050	1.0
White gaussian noise	0.9183	0.9450	0.9917	0.9333	1.0
Salt and pepper noise	0.9150	0.9367	0.9850	0.940	1.0
Speckle noise	0.9033	0.9316	1.0	0.9217	1.0
White gaussian & salt and pepper noises	0.8667	0.8917	0.9717	0.9050	0.9983
Salt and pepper & speckle noises	0.8517	0.8883	0.9750	0.8967	0.9933
Horizontal distribution of noise combinations	0.8833	0.9150	0.9817	0.920	1
Vertical distribution of noise combinations	0.890	0.9083	0.9767	0.9117	1
MPEG-2 compression	0.9517	0.9750	0.8683	0.8133	0.9816
Frame scaling	0.9729	0.9983	0.9958	0.8354	1.0
Random frame dropping	0.9722	0.8028	0.7028	0.6389	0.9750
Video rotation	0.6667	0.9944	0.9883	0.5028	0.7361

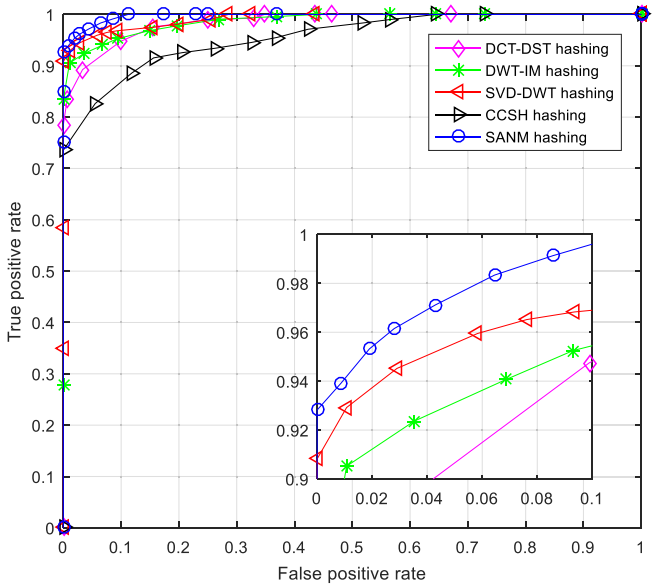

Fig. 13. ROC curve among comparison algorithms.

computed for further evaluation. The AUCs for the DCT-DST hashing, DWT-IM hashing, SVD-DWT hashing, CCSH hashing, and SANM hashing are 0.9886, 0.9912, 0.9936, 0.9502, and 0.9981, respectively. The SANM hashing method's AUC is higher than the comparison method. The outcomes demonstrate that the proposed hash approach performs better for classification than the comparison method. As a result, the performance of the proposed hashing method for classification is advanced. This is mainly because the SANM model is robust to various multiple noise attacks, and the proposed model is decomposed into a content-based representation by using SDD, which provides good discrimination ability for the SANM hashing. The complementary strengths of SANM and SDD help to improve the tradeoff between robustness and discrimination. Additionally, on the ROC graph, if two hashing methods have the same P_{FPR}, the algorithms with a high P_{TPR} outperform those with a low P_{TPR}. We computed the optimal P_{TPR} value for each hash method of comparison under various operations where P_{FPR} takes a smaller value, and the results are displayed in Table V. Our P_{TPR} all reach their maximum values, with the exception of video rotation, as can be seen. White Gaussian & salt and pepper noises, and Salt and pepper & speckle noises have PTPR of 0.9983 and 0.9933, which are very close to 1. Because of this, SANM hashing is superior than other hashing algorithms in that it can accurately detect visually similar videos and is more resistant to typical digital operations.

For complexity analysis, the computation time is adopted to measuring comparison algorithm. We initially normalize the test videos to the same 256 × 256 × 256 spatial complexity in order to conduct a fair comparison. Every source code was implemented in MATLAB and executed on a system with an Intel Core Intel Core i7-8750H running at 2.26 GHz and 16 GB of RAM. The average time for the DCT-DST hashing, DWT-IM hashing, SVD-DWT hashing, CCSH hashing, and SANM hashing are 18.23, 10.35, 15.46, 8.84, and 32.25 s, respectively. CCSH is faster than compared algorithms, this is because the approach simply involves a variety of linear computations and excludes the usage of computationally intensive transformation domains. As a result, the algorithm operates more effectively. The DCT-DST hashing, DWT-IM hashing, SVD-DWT hashing are moderate, and SANM hashing is comparatively complex. This is due to the SANM hash algorithm's reconstruction of the input video and the addition of the security computing module, which guarantees the security of the hash production, but also increases the computational complexity.

Moreover, the lengths of video hashes of different algorithms are compared. Similarly, normalized videos are also utilized for test videos in order to fairly calculate the hash lengths for different algorithms. The length of the DCT-DST hashing, DWT-IM hashing, SVD-DWT hashing, CCSH hashing, and SANM hashing are 4224 integers, 56 integers, 16 integers, 16384 bits, and 384 integers. According to the IEEE floating point arithmetic standard [49], floating point numbers must be stored using a minimum of 32 bits. After applying the IEEE standard calculation, the DCT-DST hashing, DWT-IM hashing, SVD-DWT hashing, CCSH hashing, and SANM hashing, respectively, have lengths of 135168, 1792, 512, 16384, and 12288 bits. SANM hashing uses a moderate hash length, which is a little longer than that of DWT-IM hashing and SVD-DWT hashing but much less than that of DCT-DST hashing. Results of more thorough performance comparisons are shown in Table VI.

TABLE VI
SUMMARIZES THE PERFORMANCE COMPARISON BETWEEN DIFFERENT VIDEO HASHING ALGORITHMS

Performance \ Algorithms	DCT-DST hashing	DWT-IM hashing	SVD-DWT hashing	CCSH hashing	SANM hashing
AUCs	0.9886	0.9912	0.9936	0.9502	0.9981
Average Time(s)	18.23	10.35	15.46	8.84	32.25
Storage	4224 integers	56 integers	16 integers	16384bits	384 integers

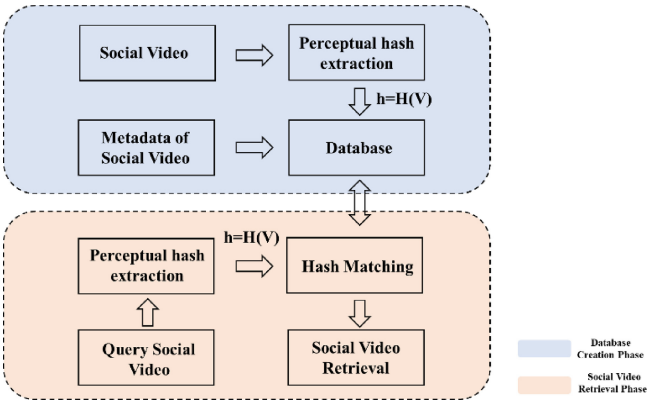

Fig. 14. Block diagram of the perceptual video hashing-based social video retrieval.

G. Social Video Retrieval

A database of 13 000 social videos was built to verify the performance of social video retrieval. In this database, 1000 social videos are used from MSVD [48]. The other 12 000 videos are content approximation versions, which are generated from 1000 source videos using 12 content retention operations. The used content-preserving operations contain brightness adjustment with the Photoshop's scale is 20, White Gaussian noise with variance is 0.1, Salt and pepper noise with Density is 0.1, Speckle noise with variance is 0.1, White Gaussian & Salt and pepper noises with hybrid parameter [0.1,0.1], Salt and pepper & speckle noises with hybrid parameter [0.1,0.1], Horizontal distribution of noise combinations with combination parameter [0.1,0.1,0.1], Vertical distribution of noise combinations with combination parameter [0.1,0.1,0.1], MPEG-2 compression with kilobit per second is 100, Frame scaling with ratio is 1.2, Random frame dropping with frame number is 20, and Video rotation with angle 2 degrees. The accuracy was estimated by counting the number of accurately detected content retention versions in the first 12 videos that provided the most similar results in order to assess the retrieval performance of different algorithms. The entire retrieval performance was also shown using the precision-recall curve (P-R curve). The best performance is indicated by the curve that is closest to the graph's upper right corner. Fig. 14 shows a social video retrieval system that uses perceptual video hashes. Fig. 15 displays the P-R curves of several hashing techniques as a visual result. The P-R curve for SANM hashing is closer to the upper right corner than the P-R curve for other algorithms. As a result, the SANM hashing approach is better than the other four hash method for retrieving social videos.

The overall social video retrieval accuracy on the database is tested, and the SANM hashing method is compared with other

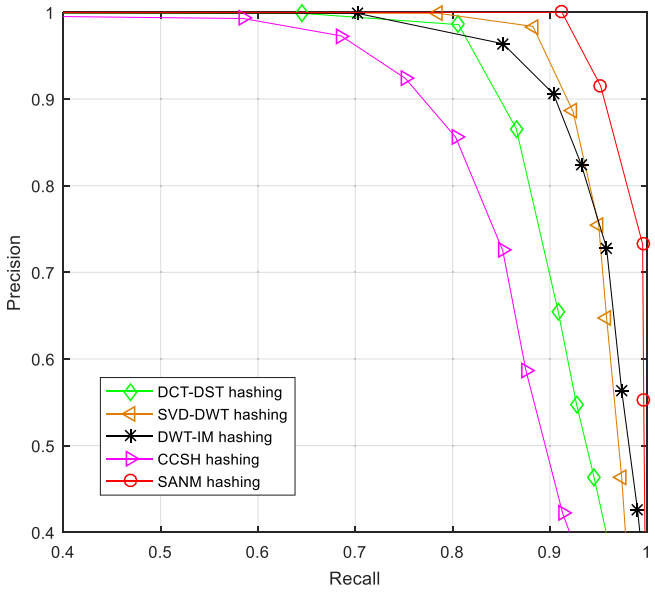

Fig. 15. P-R curves among comparison hashing.

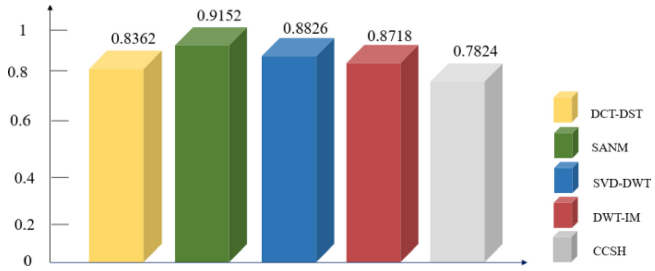

Fig. 16. Social video retrieval accuracy comparison.

video hash algorithms. To be more precise, we use the compared algorithms to extract hash sequences from social videos, compare the hash sequences of the query and test videos, and then select the first 12 videos with the highest similarity. As shown in Fig. 16, we calculated each algorithm's accuracy by counting the correct social videos among the returning videos. Evidently, DCT-DST hashing, SANM hashing, SVD-DWT hashing, DWT-IM hashing, and CCSH hashing have overall retrieval accuracy values of 0.8362, 0.9152, 0.8826, 0.8718, and 0.7824, respectively. The application effect of the suggested method is superior to that of the comparison algorithm in social video retrieval, as shown by the fact that SANM hashing has a greater accuracy than all comparison algorithms.

IV. CONCLUSION

In this article, a novel video hashing with an SANM is proposed. A robust and secure video model is built using the

low-rank component of the LRR and RSP, which gives our hash good noise resistance. To derive a compact hash with a desired discriminate, hashes formed from calculation SDD using factor matrices by Tucker decomposition and invariant distances are also useful. The outcomes of numerous experiments showed that the suggested hashing was reliable for some content-preserving operations and achieved good discrimination. The SANM hashing algorithm achieves a decent tradeoff between robustness and discrimination when compared to several popular hashing methods. Furthermore, it offers excellent precision for retrieving social video.

REFERENCES

[1] Y. Wang, X. Ou, J. Liang, and Z. Sun "Deep semantic reconstruction hashing for similarity retrieval," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 1, pp. 387–400, Jan. 2021.

[2] X. Nie, J. Qiao, J. Liu, J. Sun, X. Li, and W. Liu, "LLE-based video hashing for video identification," in *Proc. IEEE 10th Int. Conf. Signal Process.*, 2010, pp. 1837–1840.

[3] W. Tang, Y. Wo, and G. Han, "Geometrically robust video hashing based on ST-PCT for video copy detection," *Multimedia Tools Appl.*, vol. 78, no. 15, pp. 21999–22022, 2019.

[4] Z. Chen, J. Lu, J. Feng, and J. Zhou, "Nonlinear structural hashing for scalable video search," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1421–1433, Jun. 2018.

[5] Y. Dengpan, W. Zhuo, D. Xuhua, and R. H. Deng, "Scalable content authentication in H.264/SVC videos using perceptual hashing based on Dempster–Shafer theory," *Int. J. Comput. Intell. Syst.*, vol. 5, no. 5, pp. 953–963, 2012.

[6] Q. Chuan, Z. Wei, C. Fang, Z. Xinpeng, and C. Chin-Chen, "Separable reversible data hiding in encrypted images via adaptive embedding strategy with block selection," *Signal Process.*, vol. 153, pp. 109–122, Dec. 2018.

[7] A. Mucedero, R. Lancini, and F. Mapelli, "A novel hashing algorithm for video sequences," in *Proc. Int. Conf. Image Process.*, 2004, pp. 2239–2242.

[8] B. Coskun and B. Sankur, "Robust video hash extraction," in *Proc. 12th Eur. Signal Process. Conf.*, 2004, pp. 2295–2298.

[9] S. Lee and C. D. Yoo, "Robust video fingerprinting for content-based video identification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 7, pp. 983–988, Jul. 2008.

[10] X. Nie et al., "Key-frame based robust video hashing using isometric feature mapping," *J. Comput. Inf. Syst.*, vol. 7, no. 6, pp. 2112–2119, 2011.

[11] G. Yang, N. Chen, and Q. Jiang, "A robust hashing algorithm based on SURF for video copy detection," *Comput. Secur.*, vol. 31, no. 1, pp. 33–39, 2012.

[12] M. Li and V. Monga, "Robust video hashing via multilinear subspace projections," *IEEE Trans. Image Process.*, vol. 21, pp. 4397–4409, 2012.

[13] N. Saikia, "Perceptual hashing in the 3D-DWT domain," in *Proc. Int. Conf. Green Comput. Internet Things (ICGCIoT)*, 2015, pp. 694–698.

[14] R. Sandeep, S. Sharma, M. Thakur, and P. K. Bora, "Perceptual video hashing based on Tucker decomposition with application to indexing and retrieval of near-identical videos," *Multimedia Tools Appl.*, vol. 75, no. 13, pp. 7779–7797, 2016.

[15] I. Setyawan and I. K. Timotius, "Spatio-temporal digital video hashing using edge orientation histogram and discrete cosine transform," in *Proc. Int. Conf. Inf. Technol. Syst. Innov. (ICITSI)*, 2014, pp. 111–115.

[16] X. Nie, Y. Chai, J. Liu, J. Sun, and Y. Yin, "Spherical torus-based video hashing for near-duplicate video detection," *Sci. China Inf. Sci.*, vol. 59, no. 5, 2016, Art. no. 59101.

[17] R. Sandeep, S. Sharma, and P. K. Bora, "Perceptual video hashing using 3D-radial projection technique," in *Proc. 4th Int. Conf. Signal Process. Commun. Netw. (ICSCN)*, 2017, pp. 1–6.

[18] S. Rameshnath and P. K. Bora, "Perceptual video hashing based on temporal wavelet transform and random projections with application to indexing and retrieval of near-identical videos," *Multimedia Tools Appl.*, vol. 78, no. 13, pp. 18055–18075, 2019.

[19] F. Khelifi and A. Bouridane, "Perceptual video hashing for content identification and authentication," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 1, pp. 50–67, Jan. 2019.

[20] Z. Tang, L. Chen, H. Yao, X. Zhang, and C. Yu, "Video hashing with DCT and NMF," *Comput. J.*, vol. 63, no. 7, pp. 1017–1030, 2020.

[21] L. Chen, D. Ye, and Y. Shang, "RTIM hashing: Robust and compact video hashing with a rotation-and translation-invariant model," *Comput. J.*, to be published.

[22] Z. Tang, S. Zhang, X. Zhang, Z. Li, Z. Chen, and C. Yu, "Video hashing with secondary frames and invariant moments," *J. Vis. Commun. Image Represent.*, vol. 79, Aug. 2021, Art. no. 103209.

[23] Z. Chen, Z. Tang, X. Zhang, R. Sun, and X. Zhang, "Efficient video hashing based on low-rank frames," *IET Image Process.*, vol. 16, no. 2, pp. 344–355, 2022.

[24] L. Tang, Q. Ye, H. Zheng, H. Hu, Z. Han, and N.-F. Law, "Stateful-CCSH: An efficient authentication scheme for high-resolution video surveillance system," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 19373–19386, Oct. 2022.

[25] L. Chen, D. Ye, Y. Shang, and J. Huang, "Robust video hashing based on local fluctuation preserving for tracking deep fake videos," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2022, pp. 2894–2898.

[26] X. Hu et al., "Emotion-aware cognitive system in multi-channel cognitive radio ad hoc networks," *IEEE Commun. Mag.*, vol. 56, no. 4, pp. 180–187, Apr. 2018.

[27] K. Guo, L. Liu, X. Xu, D. Xu, and D. Tao, "GoDec+: Fast and robust low-rank matrix decomposition based on maximum correntropy," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2323–2336, Jun. 2018.

[28] J. Chen and J. Yang, "Robust subspace segmentation via low-rank representation," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1432–1445, Aug. 2014.

[29] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proc. ICML*, 2010, pp. 663–670.

[30] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.

[31] B. Mondal, S. Singh, and P. Kumar, "A secure image encryption scheme based on cellular automata and chaotic skew tent map," *J. Inf. Secur. Appl.*, vol. 45, pp. 117–130, Apr. 2019.

[32] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image Vis. Comput.*, vol. 24, no. 9, pp. 926–934, 2006.

[33] P. Comon, "Tensor decompositions," in *Mathematics in Signal Processing V*. Oxford, U.K.: Oxford Univ. Press, 2002, pp. 1–24.

[34] M. Ouerfelli, M. Tamaazousti, and V. Rivasseau, "Random tensor theory for tensor decomposition," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 7913–7921.

[35] J. Kim, S. Tariq, and S. S. Woo, "PTD: Privacy-preserving human face processing framework using tensor decomposition," in *Proc. 37th ACM/SIGAPP Symp. Appl. Comput.*, 2022, pp. 1021–1030.

[36] A. Cichocki et al., "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Process. Mag.*, vol. 32, no. 2, pp. 145–163, Mar. 2015.

[37] L. R. Tucker, "Implications of factor analysis of three-way matrices for measurement of change," in *Problems in Measuring Change*, vol. 15. Madison, WI, USA: Univ. Wisconsin Press, 1963, pp. 122–137.

[38] P. Shao, D. Zhang, G. Yang, J. Tao, F. Che, and T. Liu, "Tucker decomposition-based temporal knowledge graph completion," *Knowl. Based Syst.*, vol. 238, Feb. 2022, Art. no. 107841.

[39] J. Woo, W. Y. Kim, and S. Choi, "System-specific separable basis based on Tucker decomposition: Application to density functional calculations," *J. Chem. Theory Comput.*, vol. 18, no. 5, pp. 2875–2884, 2022.

[40] Z. Tang, L. Chen, X. Zhang, and S. Zhang, "Robust image hashing with tensor decomposition," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 3, pp. 549–560, Mar. 2019.

[41] L. De Lathauwer, B. De Moor, and J. Vandewalle, "On the best rank-1 and rank-$(R_1, R_2, \ldots, R_N)$ approximation of higher-order tensors," *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 4, pp. 1324–1342, 2000.

[42] Z. Huang and S. Liu, "Robustness and discrimination oriented hashing combining texture and invariant vector distance," in *Proc. 26th ACM Int. Conf. Multimedia*, 2018, pp. 1389–1397.

[43] M. Yu, Z. Tang, Z. Li, X. Liang, and X. Zhang, "Robust image hashing with saliency map and sparse model," *Comput. J.*, 2022.

[44] "VOT2015." Visual Object Tracking. 2015. Accessed: May 23, 2022. [Online]. Available: https://www.votchallenge.net/vot2015

[45] "ALOV300++." Amsterdam Library of Ordinary Videos. Accessed: May 27, 2022. [Online]. Available: http://alov300pp.joomlafree.it

[46] M. L. McHugh, "The chi-square test of independence," *Biochemia Medica*, vol. 23, no. 2, pp. 143–149, 2013.

[47] J. A. Hanley and B. J. McNeil, "A method of comparing the areas under receiver operating characteristic curves derived from the same cases," *Radiology*, vol. 148, no. 3, pp. 839–843, 1983.
[48] "MSVD." Microsoft Research Video Description. Accessed: Jun. 15, 2022. [Online]. Available: http://www.cs.utexas.edu/users/ml/clamp/videoDescription/YouTubeClips.tar
[49] M. Cornea, "IEEE 754-2008 decimal floating-point for Intel® architecture processors," in *Proc. 19th IEEE Symp. Comput. Arithmetic*, 2014, pp. 225–228.

Lv Chen received the B.S. and M.Eng. degrees from Guangxi Normal University, Guilin, China, in 2015 and 2018, respectively. He is currently pursuing the Ph.D. degree with the School of Cyber Science and Engineering, Wuhan University, Wuhan, China.

His research interests include image hashing, video hashing, video forensics, machine learning, and multimedia security.

Dengpan Ye received the B.S. degree in automatic control from the South China University of Technology, Guangzhou, China, in 1996, and the Ph.D. degree from Nanjing University of Science and Technology, Nanjing, China, in 2005.

He was a Postdoctoral Fellow with Information System, School of Singapore Management University, Singapore. Since 2012, he has been a Professor with the School of Cyber Science and Engineering, Wuhan University, Wuhan, China. He has authored or coauthored over 70 refereed journal and conference papers. His research interests include machine learning and multimedia security.

Yueyun Shang received the B.A. degree in applied mathematics and the Ph.D. degree with the School of Mathematics and Statistics, Central China Normal University, Wuhan, China, in 2002 and 2014, respectively.

Since 2015, she has been an Associate Professor with the School of Mathematics and Statistics, South-Central University, Wuhan. Her research interests include applied mathematics and multimedia security.