IJCNN 2019. International Joint Conference on Neural Networks. Budapest, Hungary. 14-19 July 2019

A Fast Feature Extraction Algorithm for Image and Video Processing

Sadiq H. Abdulhussain
Department of Computer Engineering
University of Baghdad
Baghdad, Iraq
sadiqh76@yahoo.com
sadiqhabeeb@coeng.uobaghdad.edu.iq

Abd Rahman Ramli
Computer and Communication Engineering
Universiti Putra Malaysia
Selangor, Malaysia
arr@upm.edu.my

Basheera M. Mahmmod
Department of Compute Engineering
University of Baghdad
Baghdad, Iraq
basheera412@yahoo.com
basheera.m@coeng.uobaghdad.edu.iq

M. Iqbal Saripan
Computer & Communication Engineering
Universiti Putra Malaysia
Selangor, Malaysia
iqbal@upm.edu.my

S.A.R. Al-Haddad
Computer & Communication Engineering
Universiti Putra Malaysia
Selangor, Malaysia
sar@upm.edu

Thar Baker
Department of Computer Science
Liverpool John Moores University
Liverpool, UK
t.baker@ljmu.ac.uk

Wameedh N. Flayyih
Department of Computer Engineering
University of Baghdad
Baghdad, Iraq
wam_nazar@yahoo.com

Wissam A. Jassim
ADAPT Center, School of Engineering,
Trinity College Dublin
the University of Dublin
Dublin 2, Ireland
wissam.jassim@tcd.ie

***Abstract*—Medical images and videos are utilized to discover, diagnose and treat diseases. Managing, storing, and retrieving stored images effectively are considered important topics. The rapid growth of multimedia data, including medical images and videos, has caused a swift rise in data transmission volume and repository size. Multimedia data contains useful information; however, it consumes an enormous storage space. Therefore, high processing time for that sheer volume of data will be required. Image and video applications demand for reduction in computational cost (processing time) when extracting features. This paper introduces a novel method to compute transform coefficients (features) from images or video frames. These features are used to represent the local visual content of images and video frames. We compared the proposed method with the traditional approach of feature extraction using a standard image technique. Furthermore, the proposed method is employed for shot boundary detection (SBD) applications to detect transitions in video frames. The standard TRECVID 2005, 2006, and 2007 video datasets are used to evaluate the performance of the SBD applications. The achieved results show that the proposed algorithm significantly reduces the computational cost in comparison to the traditional method.**

***Index Terms*—Block processing, Feature Extraction, Orthogonal Polynomials, Shot Boundary Detection, Temporal Video Segmentation, Abrupt Transition Detection**

I. Introduction

During the past decades, the development of computer performance, availableness of storage media, advancement of multimedia technologies, and the development of IoT resulted in large size of multimedia data. Image and video are considered the most consuming in terms of storage space; however, they contain much useful information [1].

Decades ago, databases were comparatively small in size. The annotation was performed manually using keywords. Conversely, nowadays, databases have become enormous in size, structure, and information. Accordingly, searching for information or video event within huge databases is difficult, explicitly it is a time-consuming process and requires more efforts from users [2]. Therefore, an automated video structure analysis is essential, which involves content-based video indexing and retrieval (CBVIR). CBVIR system aims to automate the management, video indexing, and retrieval [3], which are performed based on video's spatiotemporal, visual, and semantic contents [4].

A video consists of single or multiple stories [5], a story, which captures a series of events, is composed of several scenes. A scene is a pool of semantically related and temporally contiguous shots captured at various camera angles [6], [7]. Accordingly, a shot is the fundamental entity of the video sequence, and it is the entity used for the high-level indexing and retrieval tasks [1], [8]. The shot level analysis is a substantial step for semantic content representation; and thus, essential for indexing and retrieval tasks.

A shot is a contiguous sequence of frames having temporal and locale connections. Shot frames are acquired and logged by a single camera. In the video production process, shots are aggregated together to form a scene, and scenes are concatenated together to form the entire video. The aggre-

978-1-7281-2009-6/$31.00 paper N-19608.pdf

gation between video shots is known as shot transitions. Shot transitions, which are divided mainly into two types, namely, hard transition (HT) and soft transition (ST). The HT is the editing process of concatenating two shots side by side. While, the ST is the editing process of involving multiple frames in the transition and has many forms such as fade, wipe, and dissolve. Shot boundary detection (SBD) process is considered the initial and substantial stage in CBVIR system which aims to detect transitions between consecutive shots of the input videos to be used by other CBVIR modules. Feature extraction is considered the first step in SBD algorithms which aims to obtain a significant representation of visual information.

In SBD, the extraction of features is implemented either in the compressed domain (COD) or uncompressed domain (UCD). Several researchers have addressed the problems of SBD in the COD [1]. Feature extraction process entails a low computation time because it works directly in COD [9]. Nevertheless, features are not representative because they do not deal with the visual information, and are highly dependent on the COD scheme. Accordingly, the researchers are shifting their attention toward the UCD because of the amount of visual information in the frame, in addition of being more accurate than COD.

The uncompressed domain is developed from pixel-based approach [10] to encompass other approaches such as: histogram-based [11], edge-based [12], transform-based [13], and local keypoint-based [8] approaches.

In transform-based SBD algorithms, various studies used discrete transforms such as Fourier transform [14]–[17], Wavelet transform [18], Zernike transform and Fourier-Mellin transform [4], and Walsh-Hadamard transform [13], [19] as a feature extraction tool. Although these algorithms have good detection accuracy [18], they are considered computationally expensive [2]. The high computational cost stems from employing block processing for local feature extraction (transform coefficients) [2]. It is worth noting that the accuracy of SBD can be improved by extracting local features because they are considered more tolerant to a flashlight, object, and camera motion than global features [2].

In most image and video processing applications, image or video frame is partitioned into blocks. Each block is processed by a transformation function to extract features. The extracted features are stored in a memory location corresponding to the image block. Then, these features are utilized for further processing steps. Fig. 1 shows a block diagram of the traditional feature extraction process for the input image. In the traditional image block partitioning, the image blocks are extracted and then processed sequentially. However, from the memory perspective, this process is not sequential. It is known that the primary drawback of the overall computer performance is related to the big gap between CPU and memory speeds [20]. Accessing the whole matrix in a sequence has more predictable behavior from memory (especially cache) perspective as it preserves the spatial locality. This is true since the entire image matrix is usually stored in a row or column order in memory. On the other hand, dividing the matrix into blocks during

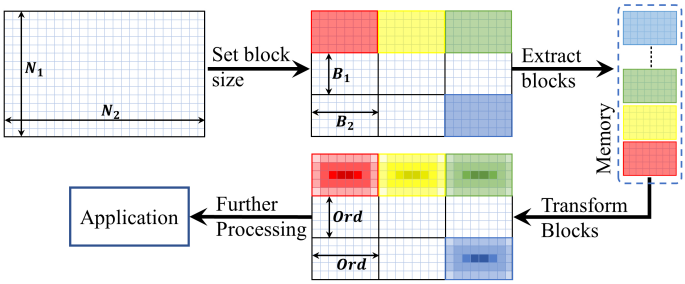

Fig. 1. Transform coefficients (features) computation using traditional block processing.

processing is translated into irregular access patterns leading to more cache misses and replacements.

Accordingly, the exclusion of the additional processes will accelerate the process of local feature extraction. These processes are the image block partitioning, sequential processing of image blocks, and the result accumulation. Motivated by this idea, we introduce a new method to extract local features from the image (video frame) blocks using discrete transform with low computational complexity.

This paper is organized as follows: Section II introduces the basic aspects of orthogonal polynomials (OPs) and the mathematical model of the implemented OP. Section III explains the proposed method for extracting local features. Section IV presents the experimental results to highlight the effectiveness of the proposed method. Finally, the last section summarizes the work with future direction of the research work.

II. Orthogonal Polynomials

A discrete transform is a useful tool in communication, signal processing, and signal representation [21], [22]. Discrete transform involves transforming a signal (frame) from the time (spatial) domain into the transform domain which is performed using the transformation function (orthogonal polynomial). It allows the viewing of signals in different domains and provides a massive shift for its powerful ability to analyze the components of various signals [1], [21].

In addition, orthogonal polynomials (OPs) have been extensively studied and used in several applications [23]. Orthogonal moments (OMs) are generated using OP [24]. OMs are scalar quantities and used as a shape descriptor because of their remarkable and efficient performance [25], [26].

Krawtchouk [25] and Tchebichef [27] polynomials are examples of OPs. The discrete Krawtchouk transform (DKT) and discrete Tchebichef transform (DTT) are constructed based on the OP kernels of the KP and TP, respectively. Recently, hybrid forms of Krawtchouk and Tchebichef polynomials have been suggested by researchers, namely Krawtchouk-Tchebichef polynomials (KTP) [23] and Tchebichef-Krawtchouk polynomials (TKP) [28]. KTP and TKP kernels are used to transform the signal from the spatial domain to the moment domain, namely discrete Krawtchouk-Tchebichef transform (DKTT) and Tchebichef-Krawtchouk transform (DTKT), respectively.

The performance of the KTP and TKP sets showed remarkable characteristics in terms of energy compaction (EC) and localization properties compared with various existing real transforms [7]. An extensive analysis was performed among DTKT, DKTT, DKT, DTT, and DCT in [23], [28]. This analysis showed that DTKT and DKTT are able to extract features from a particular portion of image and speech signals by the merits of the localization property in time and space domains. Moreover, these OPs show high EC, since they redistribute signal energy into a small number of polynomial coefficients. EC can be characterized by the fraction of the total number of signal transform coefficients that carry a certain percentage of the signal energy. Fig. 2 shows the 3D

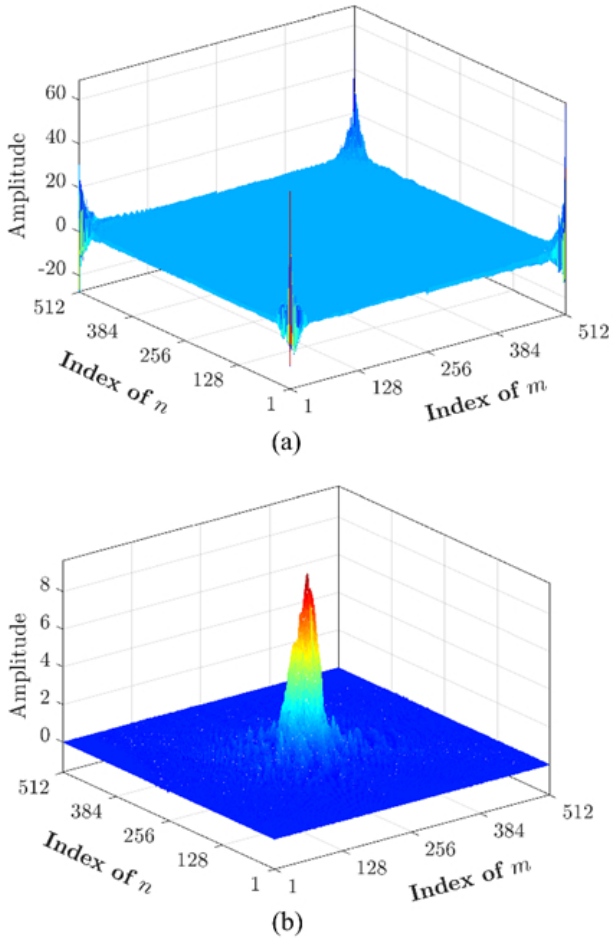

Fig. 2. The representation of the test image (Cameraman) in (a) DTKT domain, and (b) DKTT domain.

plots of the moments for the test image (Cameraman) using DTKT and DKTT. The distribution of DKTT moments is concentrated at the center of the moments array; while for DTKT the distribution is concentrated at the corners of the moments array [7]. Noticeably, the computational cost for extracting the dominant moments (features) of the test image in the DKTT is less than that of DTKT. Thus, in this paper, DKTT is used to extract features from the processed image (video frame).

In the following, we present the mathematical model of KTP. The nth order of KTP, $R_n(x)$, is given by [23]:

$$\begin{aligned} R_n(x) &= \sum_{i=0}^{N-1} k_i(n;p,N-1)\ t_i(x) \\ n,x &= 0,1,\ldots,N-1, \\ N &> 0,\ p \in (0,1) \end{aligned} \tag{1}$$

where $t_i(x)$ is the TP of the ith order which is given by [27]:

$$\begin{aligned} t_i(x) &= \frac{(1-N)_i\ {}_3F_2\left(-i,-x,1+i;1,1,1-N;1\right)}{\sqrt{(2i)!\binom{N+i}{2i+1}}} \\ i,x &= 0,1,\ldots,N-1,N>0 \end{aligned} \tag{2}$$

where $\binom{a}{b}$ is the binomial coefficients, and $(a)_k$ is the ascending factorial and is defined as follows:

$$(a)_k = a(a+1)(a+2),\ldots,(a+k-1) = \frac{\Gamma(a+k)}{\Gamma(a)} \tag{3}$$

$k_i(n;p,N-1)$ is the KP of the ith order given as [25]:

$$\begin{aligned} k_i(n;p,N-1) &= \sqrt{\frac{A}{B}}\ {}_2F_1\left(-i,-n;-N+1;1/p\right) \\ A &= \binom{N-1}{n} p^n (1-p)^{N-1-n} \\ B &= (-1)^i \left(\frac{1-p}{p}\right)^i \frac{i!}{(-N+1)_i} \\ i,x &= 0,1,\ldots,N-1, \\ N &> 0,\ p \in (0,1) \end{aligned} \tag{4}$$

Finally, ${}_3F_2$ and ${}_2F_1$ in Eqs. (2) and (4), respectively, are the hypergeometric functions given by [23]:

$${}_3F_2(-i,-x,i+i;1,1-N;1) = \sum_{j=0}^{\infty} \frac{(-i)_j\ (-x)_j\ (1+i)_j}{(1)_j\ (1-N)_j\ j!} \tag{5}$$

$${}_2F_1(-i,-n;-N+1;\frac{1}{p}) = \sum_{j=0}^{\infty} \frac{(-i)_j\ (-n)_j\ \left(\frac{1}{p}\right)^j}{(-N+1)_j\ j!} \tag{6}$$

III. Proposed Block Processing Method

This paper presents a new method of applying the transformation function on the partitioned blocks to extract features from an image or video frame directly without using the processes of the traditional method. Accordingly, the mathematical derivation details are presented as follows.

Let the image I of size $N1 \times N2$ pixels be partitioned into blocks, where the block size is $B_1 \times B_2$. The total number of blocks is $v_1 \times v_2$, where $v_1 = N_1/B_1$ and $v_2 = N_2/B_2$. The partitioned blocks of the image can be represented in matrix form as follows:

$$I = \begin{bmatrix} BL_{1,1} & BL_{1,2} & \cdots & BL_{1,v_2} \\ BL_{2,1} & BL_{2,2} & \cdots & BL_{2,v_2} \\ \vdots & \vdots & \ddots & \vdots \\ BL_{v_1,1} & BL_{v_1,2} & \cdots & BL_{v_1,v_2} \end{bmatrix} \tag{7}$$

where, $BL_{i,j}$ refers to the image block in the i- and j-directions and it is defined as follows:

$$\begin{aligned} BL_{i,j} &= I(x,y) \\ x &= (j-1)B_2, (j-1)B_2-1, \ldots, jB_2-1 \\ y &= (i-1)B_1, (i-1)B_1-1, \ldots, iB_1-1 \end{aligned} \tag{8}$$

For more clarification, Fig. 3 shows the image blocks indices. In the traditional method, after partitioning the image into

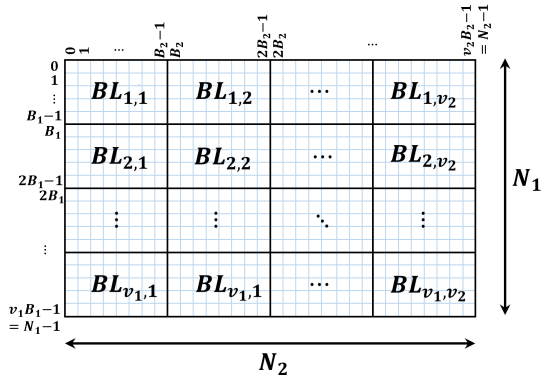

Fig. 3. The image block indices.

blocks, the transformation function is applied for each block using the following formula:

$$\begin{aligned} \Psi_{BL_{i,j}} &= R_{B_1} BL_{i,j} R_{B_2}^T \\ i &= 1, 2, \ldots, v_1, \\ j &= 1, 2, \ldots, v_2 \end{aligned} \tag{9}$$

where $R_{B_1} \in \mathbb{R}^{Ord \times B_1}$ and $R_{B_2} \in \mathbb{R}^{Ord \times B_2}$ are the transform matrices which represent the transformation functions, $(\cdot)^T$ refers to matrix transpose, and Ord is the order of the transform which represents the number of selected features used for visual content representation and image reconstruction ($1 \leq Ord \leq B_1$). Note that, $\Psi_{BL_{i,j}} \in \mathbb{R}^{Ord \times Ord}$ is the matrix of the transform coefficients (features) computed for the individual image block, $BL_{i,j}$. In other words, $\Psi_{BL_{i,j}}$ represents the image block $BL_{i,j}$ in the transform domain. The image block can be reconstructed from the selected features using:

$$\widetilde{BL}_{i,j} = R_{B_1}^T \Psi_{BL_{i,j}} R_{B_2} \tag{10}$$

This procedure is sequentially repeated for $v_1 \times v_2$ times corresponding to the number of blocks in the image (I), which is not efficient from speed and computational cost perspective.

To overcome this problem, we propose a new method to extract features from all image blocks using one set of matrix multiplication ($P_1 \; I \; P_2^T$). The transform coefficients of all image blocks can be combined in a single matrix such that:

$$\Psi = \begin{bmatrix} R_{B_1} BL_{1,1} R_{B_2}^T & R_{B_1} BL_{1,2} R_{B_2}^T & \cdots & R_{B_1} BL_{1,v_2} R_{B_2}^T \\ R_{B_1} BL_{2,1} R_{B_2}^T & R_{B_1} BL_{2,2} R_{B_2}^T & \cdots & R_{B_1} BL_{2,v_2} R_{B_2}^T \\ \vdots & \vdots & \ddots & \vdots \\ R_{B_1} BL_{v_1,1} R_{B_2}^T & R_{B_1} BL_{v_1,2} R_{B_2}^T & \cdots & R_{B_1} BL_{v_1,v_2} R_{B_2}^T \end{bmatrix} \tag{11}$$

where $\Psi \in \mathbb{R}^{Ord \cdot v_1 \times Ord \cdot v_2}$ is the transform coefficients corresponding to the entire image blocks. It can be noticed from Eq. (11) that each block ($BL_{i,j}$) is multiplied by the two orthogonal polynomials R_{B_1} and R_{B_2}. Thus, Using Kronecker product, Eq. (11) can be rewritten as:

$$\begin{aligned} \underset{[Ord \cdot v1 \times Ord \cdot v2]}{\Psi} &= \left(\underset{[v1 \times v1]}{\mathbb{I}} \otimes \underset{[Ord \times B1]}{R_{B_1}} \right) \times \left(\underset{[N_1 \times N_2]}{I} \right) \\ &\times \left(\underset{[v2 \times v2]}{\mathbb{I}} \otimes \underset{[Ord \times B2]}{R_{B_2}^T} \right) \end{aligned} \tag{12}$$

where $\mathbb{I}$ represents the identity matrix, and $\otimes$ is the Kronecker product. Using the transpose property of the Kronecker product [29], Eq. (12) can be written as follows:

$$\underset{[Ord \cdot v1 \times Ord \cdot v2]}{\Psi} = \underset{[Ord \cdot v1 \times v1 \cdot B1]}{(\mathbb{I} \otimes R_{B_1})} \times \underset{[N_1 \times N_2]}{(I)} \times \underset{[v2 \cdot B2 \times Ord \cdot v2]}{(\mathbb{I} \otimes R_{B_2})^T} \tag{13}$$

$$= \underset{[Ord \cdot v1 \times N_1]}{P_{B_1}} \times \underset{[N_1 \times N_2]}{I} \times \underset{[N2 \times Ord \cdot v2]}{P_{B_2}^T} \tag{14}$$

where $P_{B_1} = (\mathbb{I} \otimes R_{B_1}) \in \mathbb{R}^{Ord \cdot v1 \times N1}$ and $P_{B_2} = (\mathbb{I} \otimes R_{B_2}) \in \mathbb{R}^{Ord \cdot v2 \times N2}$. The two matrices P_{B_1} and P_{B_2} are constructed from R_{B_1} and R_{B_2}, respectively. These matrices can be written as follows:

$$P_{B_1} = \underbrace{\left[\begin{array}{c|c|c|c} R_{B_1} & \mathbf{O} & \cdots & \mathbf{O} \\ \hline \mathbf{O} & R_{B_1} & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \mathbf{O} \\ \hline \mathbf{O} & \cdots & \mathbf{O} & R_{B_1} \end{array}\right]}_{N1} \Bigg\} Ord \cdot v1 \tag{15}$$

$$P_{B_2} = \underbrace{\left[\begin{array}{c|c|c|c|c} R_{B_2} & \mathbf{O} & \cdots & \cdots & \mathbf{O} \\ \hline \mathbf{O} & R_{B_2} & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & R_{B_2} & \mathbf{O} \\ \hline \mathbf{O} & \cdots & \cdots & \mathbf{O} & R_{B_2} \end{array}\right]}_{N2} \Bigg\} Ord \cdot v2 \tag{16}$$

It is noteworthy mentioning that the matrix P_B is an orthogonal matrix and satisfies the orthogonality condition [22]:

$$\begin{aligned} \sum_{x \in X} P_B(n,x) \; P_B(m,x) &= \delta_{nm} \\ n, m &= 1, 2, \ldots, Ord \end{aligned} \tag{17}$$

where δ_{nm} is the Kronecker delta [30] and symbolizes the orthonormal representation of orthogonal polynomials, and is defined by:

$$\delta_{nm} = \begin{cases} 1 & n = m \\ 0 & n \neq m \end{cases} \tag{18}$$

Because P_{B_1} and P_{B_2} are orthogonal matrices, they can be used to reconstruct the image from transform domain as follows:

$$\tilde{I} = P_{B_1}^T \; \Psi \; P_{B_2} \tag{19}$$

From Eq. (14), it can be observed that the computation of features is achieved for each distinct block $BL_{i,j}$ without partitioning the image into blocks, and using one set of matrix multiplication. Accordingly, the computational cost of the local feature extraction can be reduced. On the other hand, the computation of local features using the traditional method requires 1) partitioning the image into blocks, 2) $v_1 \times v_2$ multiplication sets using Eq. (9), and 3) accumulating the results. Whereas the proposed method does not need for image partitioning, nor accumulating the result, and it requires only one set of matrix multiplication.

Note that the proposed method is transform independent. Consequently, any type of transformation function, i.e. orthogonal polynomials, can be implemented using the proposed method.

IV. Experimental Results

In this study, features are extracted from the image or video frames using transform matrices (R_{B_1} and R_{B_2}) generated by a most recent transform which is DKTT [23]. As previously mentioned, DKTT achieved remarkable results in terms of localization and energy compaction properties for signal representation against existing discrete transforms. The transform coefficients (signal representation in the transform domain) computed by an orthogonal-polynomial-based transform are referred to as moments [23], [25], [27]. The low-order moments contain most energy of the signal, whereas the higher-order moments contain the signal details. Note that the moments order is discriminable mainly by the OP. For instance, the moments order for TP and KP is $n = 0, 1, , Ord$. While the moments order for the KTP is $n = [\frac{N}{2} - 1, \frac{N}{2}, \frac{N}{2} - 2, \frac{N}{2} + 1, \ldots, \frac{N}{2} - \frac{Ord}{2}, \frac{N}{2} + \frac{Ord}{2} - 1]$.

TABLE I
Average computation time in milliseconds for different image sizes

Image Size	Block Size $B_1 \times B_2$	Traditional Method			Proposed Method		
		Ratio of selected moments			Ratio of selected moments		
		100%	50%	25%	100%	50%	25%
256 × 256	4×4	75.03	74.38	73.91	1.78	0.72	0.33
	8×8	22.22	20.38	19.40	1.78	0.70	0.32
	16×16	7.35	6.59	6.11	1.76	0.72	0.35
512 × 512	4×4	298.99	296.81	295.89	15.47	5.64	2.85
	8×8	87.78	77.75	75.92	15.17	5.41	2.28
	16×16	26.10	23.50	21.52	15.36	5.74	2.31
1024 × 1024	4×4	1200.4	1184.3	1178.3	105.5	41.86	18.23
	8×8	347.07	312.70	311.88	103.5	41.89	18.01
	16×16	106.56	93.99	84.59	104.9	41.53	18.11

A. Speed and Computational Cost Comparison

The performance of the proposed method expressed in Eq. (14) was compared to that of the traditional method (Eq. (9)). The grayscle "Cameraman" image with a size of 256×256 pixels was employed for this experiment. Different block sizes were adopted: $B_1 = B_2 = 4, 8,$ and 16. Moments were extracted from each block for different ratios of moment selection ($\frac{Ord}{N}$): 25%, 50%, and 100%. This is controlled by the transform order, Ord. For example, when $B_1 = B_2 = 32$ and only 50% of moments are sought for block representation, Ord value is (50×32)/100=16 so that the DKTT transformation matrices R_{B_1} and R_{B_2} are generated with a size of 16×32 using the KTP [23]. In this case, there are 64 blocks as $v_1 = v_2 = 256/32 = 8$. For the proposed method, the R_{B_1} and R_{B_2} are padded with zeros to construct the corresponding P_{B_1} and P_{B_2} matrices as in Eqs. (15) and (16). As a result, the moment matrix (Ψ) with a size of 128×128 is generated. Each distinct block in the Ψ matrix represents the moments of the corresponding image block. Note that the size of the image block in the spatial domain is 32×32, and the size of their correspondence in the moments domain is 16×16. This is because the ratio of the selected moments $\frac{Ord}{N} = 50\%$. On the other hand, for the traditional method, the moments are computed for each distinct image block using Eq. (9). In other words, to compute the moments for all blocks, Eq. (9) is utilized 64 times which corresponds to the total number of blocks in the image.

The experiment is repeated for 100 times, and the averaged runtime for the proposed and traditional methods is reported. The time required for generating transform matrices for both methods is neglected. The implementation was performed using MATLAB environment on HP pavilion dv6 laptop system core i7-2670QM CPU and 8GB of RAM.

TABLE I shows the average execution time in milliseconds (ms) for different block sizes and ratios of selected features. The proposed method requires less execution time than the traditional method for all cases, especially for small block sizes. The computation time of the proposed method is approximately constant, because the image block partitioning and the results accumulation processes are omitted and only the transformation from the spatial domain to the transform domain is performed. It is clear that the number of processes is reduced dramatically; as a result, the processing runtime is decreased. On the other hand, the runtime of the traditional method is rapidly decreased as the block size increases because of the previously mentioned processes are not omitted. First, the image is partitioned into blocks. Then, each image block is transformed to the moment domain. Finally, the result is stored in a matrix. Moreover, for both methods, the execution time is affected by the ratio of selected moments to the total number of moments. For example, the time required to compute 50% of moments is less than that of 100% (full moment set). Notably, the proposed method runtime is reduced by $\sim$ 59% when the ratio of moment order is reduced from 100% to 50%. While for the traditional method, the runtime is reduced by $\sim$ 1-10%. In addition, when the ratio of moment selection is reduced from 100% to 25%, the runtime of the proposed method is reduced by $\sim$81%. Whereas, the runtime for the traditional method is reduced by $\sim$2-17%. Indeed, when the ratio of selected moments is reduced, the proposed method shows a high reduction in the runtime compared to the traditional method.

To show the effectiveness of the proposed method, the Cameraman image was resized to 512×512 and 1024×1024 pixels, and the same experiment is conducted. The achieved results are reported in TABLE I. The reported results show

that the proposed method outperforms the traditional method for different images size, block size, and ratio of selected moments.

B. Shot Boundary Detection based on the proposed method

In this study, the proposed method is employed for SBD application. SBD is implemented to detect the hard (abrupt) transition and non-transition events between consecutive shots of an input video. The well-known TRECVID2005, TRECVID2006, and TRECVID2007 datasets [31] were used for evaluation.

The moments matrix (Ψ) using KTP [23] is first constructed for each video frame to represent the frame's visual content for a predefined region-of-interest (ROI) [7]. The city-block distance metric is then computed to measure the dissimilarity signal (DS) between the moments of consecutive frames f_k and f_{k+1} as follows:

$$DS(k) = \sum_{i=1}^{Ord \cdot v_1} \sum_{j=1}^{Ord \cdot v_2} \left| \Psi_{f_k}(i,j) - \Psi_{f_{k+1}}(i,j) \right| \qquad (20)$$
$$k = 1, \ldots, M-1$$

where M is the total number of video frames. As a result, the dissimilarity signal is a vector of length $(M-1)$. A feature vector (FV) is then formed from the dissimilarity signal (DS) by considering the contextual (temporal) information [7]. The feature vector (FV) of the kth frame (f_k) is formed by the two previous, two next, and current dissimilarity signals as follows:

$$FV(k) = [DS(k-2), DS(k-1), DS(k), DS(k+1), DS(k+2)]^T \qquad (21)$$

In this case, there are 5 features for each frame. As a result, the size of the feature vector (FV) is $5 \times M - 1$. Machine learning is utilized to predict activities (labels) and to forecast time series data [32], [33]. In this work, the extracted features are then trained using support vector machine (SVM) classifier [34] to predict the two classes (hard transition and non-transition) based on the ground-truth labels of the datasets. In the training stage, the features are extracted from seven videos. These features were used to train the SVM model by utilizing ensemble learning [35]. In the testing stage, features extracted from the test videos were used as an input to the trained SVM model to predict the transition event. Note that, the size of the video frames ROI is $160 \times frame\ width$ considered from the center of the frame. Fig. 4 shows the block diagram of the implemented procedure for SBD using the proposed block processing. The experiment was performed using different values of $v_1 \times v_2$ and ratios of selected moments. TABLE II reports the computation cost (runtime) of the testing stage for both the traditional method (TM) and proposed method (PM). TABLE II presents the reported runtime in two forms: 1) the runtime for all the frames for each dataset (computation cost/dataset), and 2) the runtime required to process each frame. From the results in TABLE II, it can be observed that the proposed method reduces the computational cost of the SBD algorithm compared to the traditional method. As the number of blocks increased, the proposed method shows a comparable computational runtime when the block size increased. In addition, the proposed method exhibits a slight increase in the runtime when the ratio of the selected moments increased from 10% to 20%. Whereas the runtime consumed by the traditional method is exponentially increased as the number of blocks $v_1 \times v_2$ increased. In addition, the reported runtime shows a noticeable increase as the ratio of the selected moments increased from 10% to 20%.

TABLE II
THE COMPUTATION TIME FOR SBD BASED ON THE PROPOSED AND TRADITIONAL METHODS

Dataset	Block $v_1 \times v_2$	Ratio of Selected Moments %	Computation cost/dataset (s)		Computation cost/frame(ms)	
			TM	PM	TM	PM
2005	4×4	10	1286	723	2.185	1.229
	4×4	20	1343	773	2.282	1.314
	8×8	10	1841	712	3.129	1.210
	8×8	20	2045	789	3.475	1.341
	16×16	10	4388	722	7.457	1.227
	16×16	20	4429	793	7.527	1.348
2006	4×4	10	1023	533	2.275	1.186
	4×4	20	1066	580	2.371	1.290
	8×8	10	1533	540	3.410	1.201
	8×8	20	1641	583	3.650	1.297
	16×16	10	3353	558	7.458	1.241
	16×16	20	3384	590	7.527	1.312
2007	4×4	10	1382	781	2.398	1.355
	4×4	20	1437	852	2.493	1.478
	8×8	10	2008	778	3.484	1.350
	8×8	20	2106	857	3.654	1.487
	16×16	10	4298	798	7.457	1.384
	16×16	20	4338	861	7.526	1.494

Moreover, the accuracy measures of the SBD are reported in TABLE III. The accuracy measures include precision ($\mathcal{P}$), recall ($\mathcal{R}$), and $F1$-score. These measures are computed as follows [1]:

$$\mathcal{P} = \frac{N_{Correct}}{N_{Correct} + N_{False}} \qquad (22)$$

$$\mathcal{R} = \frac{N_{Correct}}{N_{Correct} + N_{Missed}} \qquad (23)$$

$$F1 = \frac{2\ \mathcal{P}\ \mathcal{R}}{\mathcal{P} + \mathcal{R}} \qquad (24)$$

where $N_{Correct}$ is the correctly detected transitions, N_{False} is the falsely detected transitions, and N_{Missed} is the transitions that are not detected. The results reported in TABLE III show that when the number of blocks $v_1 \times v_2$ increased the accuracy measures is increased. For example, for TRECVID2005 dataset and 10% of moments were sought, $F1$-score increases from 96.08 to 96.41 as the number of block increases from 4×4 to 16×16. The same observation can be obtained for the other datasets. This proves that when the number of blocks is increased, the effect of the disturbance factors (object and camera motion) is reduced. To show the advantage of the proposed method, a comparison has been performed between the SBD based on the proposed block processing method and

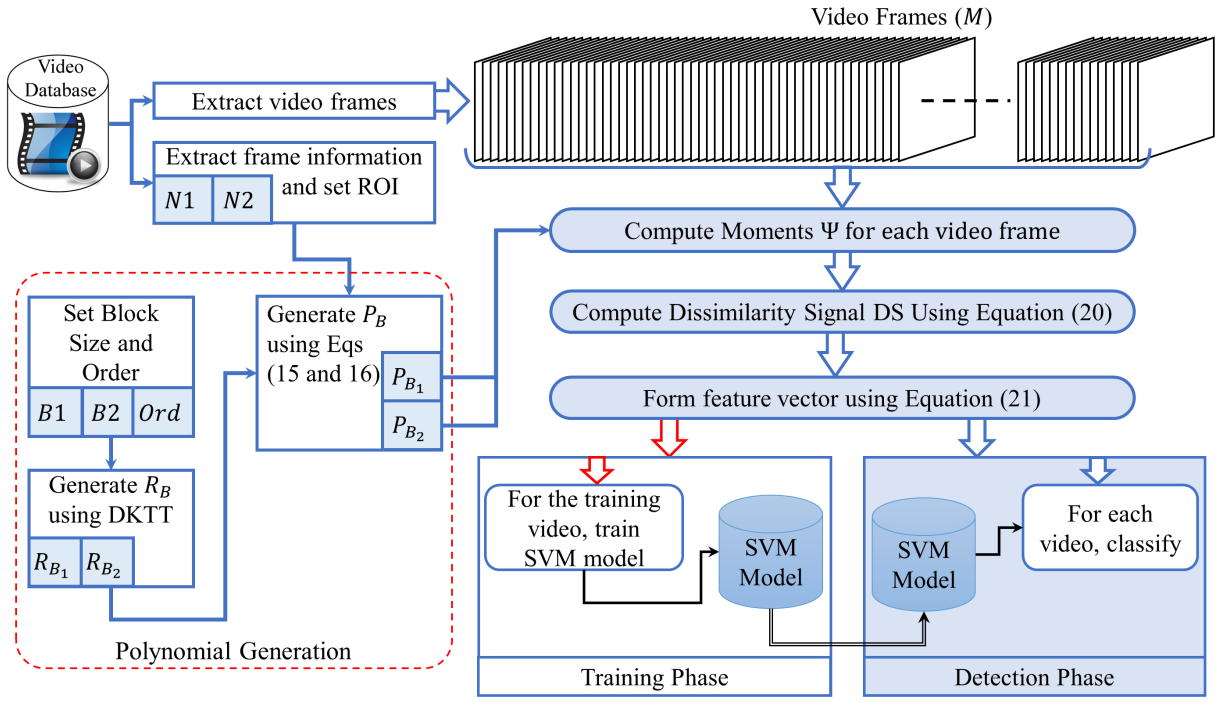

Fig. 4. The block diagram for SBD using the proposed block processing.

TABLE III
THE ACCURACY MEASURES OF SBD

Dataset	Block $v_1 \times v_2$	Ratio of Selected Moments %	Accuracy,Measure		
			P%	R%	F1%
2005	4×4	10	95.18	96.99	96.08
	4×4	20	94.93	96.69	95.80
	8×8	10	95.47	97.29	96.37
	8×8	20	95.59	97.37	96.47
	16×16	10	95.51	97.33	96.41
	16×16	20	95.76	97.54	96.64
2006	4×4	10	93.18	96.28	94.70
	4×4	20	92.91	95.71	94.28
	8×8	10	93.14	96.54	94.81
	8×8	20	93.09	95.90	94.47
	16×16	10	93.66	96.60	95.11
	16×16	20	93.26	96.73	94.97
2007	4×4	10	97.15	96.63	96.89
	4×4	20	96.81	96.38	96.59
	8×8	10	97.39	96.87	97.13
	8×8	20	96.62	96.38	96.50
	16×16	10	97.54	96.97	97.25
	16×16	20	96.91	96.58	96.74

the state-of-the-art-algorithms. These algorithms are Walsh-Hadamard transform-based SBD (WHT-SBD) [13] and non-sub sampled Contourlet Transform-based SBD (NSCT-SBD) [18]. For the SBD based on the proposed method, the block size and the ratio of selected moments are selected as follows: $v_1 = v_2 = 16$ and 10%, respectively. TABLE IV reports the runtime, precision, recall, and F1 measures, of the test videos from the TRECVID2007 dataset. TABLE IV shows that the SBD based on the proposed block processing and KTP has less computational cost than the state-of-the-art algorithms which proves the concept of the proposed block processing method. Notably, the presented SBD algorithm is $\sim$ 64 and 95 times faster than WHT-SBD and NSCT-SBD algorithms, respectively. However, the presented SBD shows slightly less accuracy than the WHT-SBD and NSCT-SBD algorithms. This is due to the fact that the presented algorithm used only one feature that is the moments of the KTP. While the WHT-SBD [13] used four types of features which are: color, edge, texture, and motion strength and motion compensation algorithm. For the NSCT-SBD [18], several decomposition levels (one for low-frequency subband and seven for high-frequency subband) were employed for each of the color space in CIE L*a*b*. This will surely increase the accuracy at the cost of the algorithm complexity. Since the accuracy measures of the presented algorithm are comparable to that of the state-of-the-art algorithms and highly reduced the computational cost; thus, the presented SBD based on the proposed method for local feature extraction outperforms the state-of-the-art algorithms.

TABLE IV
COMPUTATIONAL COST AND ACCURACY COMPARISON USING TRECVID2007 DATASET

Algorithm	Computation cost/frame (ms)	Accuracy Measures		
		$\mathcal{P}$%	$\mathcal{R}$%	$F1$%
Proposed	1.5	96.91	96.58	96.74
WHT-SBD [13]	96.63	97.42	97.79	97.61
NSCT-SBD [18]	142.48	96.36	97.66	97.01

V. CONCLUSION

This paper presented a new method for computing local features from image and video frames. The performance of the

new method is confirmed by two experiments. The first experiment is performed using "cameraman" image with different block and image sizes. The second experiment is carried out on SBD system to detect HTs. The experimental results show that the proposed method affords an exceptional computational cost improvement. Accordingly, it is beneficial for computer vision applications. Despite the superior achievement of the proposed method, the runtime can be further improved. P_{B_1} and P_{B_2} matrices can be considered as sparse or nearly sparse matrices because they are not 100% full matrices as most of their entries are zero. This property is advantageous to accelerate the process since the multiplication of a sparse and a full matrix is faster than the multiplication of two full matrices. Therefore, the execution time of Eq. (14) can be further reduced if a proper technique of sparse matrix multiplication is adopted. Considering the accuracy of the SBD based on the proposed method, our aim is to include more features to improve the accuracy without affecting the computational complexity.

REFERENCES

[1] S. H. Abdulhussain, A. R. Ramli, M. I. Saripan, B. M. Mahmmod, S. Al-Haddad, and W. A. Jassim, "Methods and Challenges in Shot Boundary Detection: A Review," *Entropy*, vol. 20, no. 4, p. 214, 2018.

[2] S. Tippaya, S. Sitjongsataporn, T. Tan, M. M. Khan, and K. Chamnongthai, "Multi-Modal Visual Features-Based Video Shot Boundary Detection," *IEEE Access*, vol. 5, pp. 12 563–12 575, 2017.

[3] R. Priya and T. N. Shanmugam, "A comprehensive review of significant researches on content based indexing and retrieval of visual information," *Frontiers of Computer Science*, vol. 7, no. 5, pp. 782–799, 2013.

[4] G. C. Chaves, "Video Content Analysis by Active Learning," PhD Thesis, PhD Thesis, Federal University of Minas Gerais, 2007.

[5] H. Bhaumik, S. Bhattacharyya, M. D. Nath, and S. Chakraborty, "Hybrid soft computing approaches to content based video retrieval: A brief review," *Applied Soft Computing*, vol. 46, pp. 1008–1029, 2016.

[6] C. Liu, D. Wang, J. Zhu, and B. Zhang, "Learning a Contextual Multi-Thread Model for Movie/TV Scene Segmentation," *IEEE Transactions on Multimedia*, vol. 15, no. 4, pp. 884–897, jun 2013.

[7] S. H. Abdulhussain, A. R. Ramli, B. M. Mahmmod, M. I. Saripan, S. A. R. Al-Haddad, and W. A. Jassim, "Shot boundary detection based on orthogonal polynomial," *Multimedia Tools and Applications*, pp. 1–22, feb 2019. [Online]. Available: http://link.springer.com/10.1007/s11042-019-7364-3

[8] M. Birinci and S. Kiranyaz, "A perceptual scheme for fully automatic video shot boundary detection," *Signal Processing: Image Communication*, vol. 29, no. 3, pp. 410–423, 2014.

[9] W. Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, "A Survey on Visual Content-Based Video Indexing and Retrieval," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 797–819, 2011.

[10] C. W. Ngo, T. C. Pong, and R. T. Chin, "Video partitioning by temporal slice coherency," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 8, pp. 941–953, 2001.

[11] Z. Li, X. Liu, and S. Zhang, "Shot Boundary Detection based on Multilevel Difference of Colour Histograms," in *2016 First International Conference on Multimedia and Image Processing (ICMIP)*, 2016, pp. 15–22.

[12] R. Zabih, J. Miller, and K. Mai, "A feature-based algorithm for detecting and classifying production effects," *Multimedia systems*, vol. 7, no. 2, pp. 119–128, 1999.

[13] L. G. G. Priya and S. Domnic, "Walsh Hadamard Transform Kernel-Based Feature Vector for Shot Boundary Detection," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 23, no. 12, pp. 5187–5197, 2014.

[14] S. V. Porter, M. Mirmehdi, and B. T. Thomas, "Video cut detection using frequency domain correlation," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 3. IEEE, 2000, pp. 409–412.

[15] T. Vlachos, "Cut detection in video sequences using phase correlation," *IEEE signal processing letters*, vol. 7, no. 7, pp. 173–175, 2000.

[16] S. Porter, M. Mirmehdi, and B. Thomas, "Temporal video segmentation and classification of edit effects," *Image and Vision Computing*, vol. 21, no. 13, pp. 1097–1106, 2003.

[17] O. Urhan, M. K. Gullu, and S. Erturk, "Modified phase-correlation based robust hard-cut detection with application to archive film," *IEEE transactions on circuits and systems for video technology*, vol. 16, no. 6, pp. 753–770, 2006.

[18] J. Mondal, M. K. Kundu, S. Das, and M. Chowdhury, "Video shot boundary detection using multiscale geometric analysis of nsct and least squares support vector machine," *Multimedia Tools and Applications*, vol. 77, no. 7, pp. 8139–8161, apr 2017.

[19] G. L. Priya and S. Domnic, "Edge Strength Extraction using Orthogonal Vectors for Shot Boundary Detection," *Procedia Technology*, vol. 6, pp. 247–254, 2012.

[20] F. Alted, "Why modern CPUs are starving and what can be done about it," *Computing in Science & Engineering*, vol. 12, no. 2, pp. 68–71, 2010.

[21] B. M. Mahmmod, A. R. Ramli, S. H. Abdulhussain, S. A. R. Al-Haddad, W. A. Jassim, S. A. R. Al-Haddad, and W. A. Jassim, "Low-Distortion MMSE Speech Enhancement Estimator Based on Laplacian Prior," *IEEE Access*, vol. 5, no. 1, pp. 9866–9881, 2017.

[22] S. H. Abdulhussain, A. R. Ramli, B. M. Mahmmod, S. A. R. Al-Haddad, and W. A. Jassim, "Image Edge Detection Operators based on Orthogonal Polynomials," *International Journal of Image and Data Fusion*, vol. 8, no. 3, pp. 293–308, 2017.

[23] B. M. Mahmmod, A. R. bin Ramli, S. H. Abdulhussain, S. A. R. Al-Haddad, and W. A. Jassim, "Signal compression and enhancement using a new orthogonal-polynomial-based discrete transform," *IET Signal Processing*, vol. 12, no. 1, pp. 129–142, aug 2018.

[24] S. H. Abdulhussain, A. R. Ramli, B. M. Mahmmod, I. Saripan, S. A. R. Al-Haddad, and W. A. Jassim, "A New Hybrid form of Krawtchouk and Tchebichef Polynomials: Design and Application," *Journal of Mathematical Imaging and Vision*, 2018.

[25] S. H. Abdulhussain, A. R. Ramli, S. A. R. Al-Haddad, B. M. Mahmmod, and W. A. Jassim, "Fast Recursive Computation of Krawtchouk Polynomials," *Journal of Mathematical Imaging and Vision*, vol. 60, no. 3, pp. 285–303, mar 2018.

[26] B. M. Mahmmod, "A Speech Enhancement Framework Using Discrete Krawtchouk-Tchebichef Transform," PhD, Universiti Putra Malaysia, 2018.

[27] S. H. Abdulhussain, A. R. Ramli, S. A. R. Al-Haddad, B. M. Mahmmod, and W. A. Jassim, "On Computational Aspects of Tchebichef Polynomials for Higher Polynomial Order," *IEEE Access*, vol. 5, no. 1, pp. 2470–2478, 2017.

[28] W. A. Jassim, P. Raveendran, and R. Mukundan, "New orthogonal polynomials for speech signal and image processing," *IET Signal Processing*, vol. 6, no. 8, pp. 713–723, 2012.

[29] H. Zhang and F. Ding, "On the Kronecker products and their applications," *Journal of Applied Mathematics*, vol. 2013, 2013.

[30] S. M. Imran, S. M. M. Rahman, and D. Hatzinakos, "Differential components of discriminative 2D GaussianHermite moments for recognition of facial expressions," *Pattern Recognition*, vol. 56, pp. 100–115, 2016.

[31] TRECVID, "TRECVID," 2016. [Online]. Available: http://trecvid.nist.gov/

[32] A. J. Hussain, D. Al-Jumeily, H. Al-Askar, and N. Radi, "Regularized dynamic self-organized neural network inspired by the immune algorithm for financial time series prediction," *Neurocomputing*, vol. 188, pp. 23–30, 2016.

[33] A. A. Mahdi, A. J. Hussain, and D. Al-Jumeily, "Adaptive Neural Network Model Using the Immune System for Financial Time Series Forecasting," in *2009 International Conference on Computational Intelligence, Modelling and Simulation*, sep 2009, pp. 104–109.

[34] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[35] M. Vij, V. Naik, and V. Gunturi, "Using smartphone-based accelerometer to detect travel by metro train," Ph.D. dissertation, 2016.