Efficient Foreground Extraction From HEVC Compressed Video for Application to Real-Time Analysis of Surveillance 'Big' Data

Bhaskar Dey and Malay K. Kundu, *Senior Member, IEEE*

***Abstract*— While surveillance video is the biggest source of unstructured Big Data today, the emergence of high-efficiency video coding (HEVC) standard is poised to have a huge role in lowering the costs associated with transmission and storage. Among the benefits of HEVC over the legacy MPEG-4 Advanced Video Coding (AVC), is a staggering 40 percent or more bitrate reduction at the same visual quality. Given the bandwidth limitations, video data are compressed essentially by removing spatial and temporal correlations that exist in its uncompressed form. This causes compressed data, which are already de-correlated, to serve as a vital resource for machine learning with significantly fewer samples for training. In this paper, an efficient approach to foreground extraction/segmentation is proposed using novel spatio-temporal de-correlated block features extracted directly from the HEVC compressed video. Most related techniques, in contrast, work on uncompressed images claiming significant storage and computational resources not only for the decoding process prior to initialization but also for the feature selection/extraction and background modeling stage following it. The proposed approach has been qualitatively and quantitatively evaluated against several other state-of-the-art methods.**

***Index Terms*— Background subtraction, HEVC, statistical signal processing, video surveillance, transform coding.**

I. Introduction

THE computational approach to video analysis is important to a large number of applications in diverse disciplines, which includes traffic monitoring, visual tracking, surveillance, video-annotation and summarization, as well as actions and gesture recognition. The first stage of analysis originates from detecting motion activities or changes in the scene. Indeed, for many applications, the very fact that something is moving makes it of 'interest' while anything else can be ignored. In such cases, it is common for regions of interest (ROI) to be categorized as *foreground* while the remaining part of the scene as *background*. The simplest approach, commonly referred to as *background subtraction* or *foreground extraction*, which involves subtracting the current frame from a model of its background. As *a priori* knowledge of a scene's background does not often exist, the key is how to learn and model it.

As stated in [1], it is difficult to specify a 'gold-standard' definition of what a background subtraction algorithm should detect as foreground, as the definition of the foreground relates to the application level. However, a majority of the applications require foreground detection algorithms to address three key issues. First, it must be robust against changes in illumination and avoid detecting shadows cast by moving objects. Second, objects of interest often move amidst complicated backgrounds that are themselves moving, e.g. swaying trees, shimmering waves, fountain, camera jitter, and snow, rain, or smoke-filled environments. Therefore, it should be able to incorporate the non-stationary entities into the background model. Last, but by no means least, the foreground detection rate should be fast enough to support real-time applications.

Recently, a new challenge has emerged in this field. Digital video has become ubiquitous in our everyday lives; everywhere we look, there are devices that can capture, encode, and transmit video. Examples of these cutting-edge systems can be found just about anywhere including airports, hospitals, automated teller machine (ATM) and banking sites, casinos, malls, retail stores, elevators, parking lots, classrooms, courts, along road-sides for traffic-violation detection, as well as those installed indoors for caring for kids or seniors. The sheer volume of video data continuously streaming into cyberspace is overwhelming, and growing by the moment. To put the challenge in perspective, the arrival of the Internet, together with the near-universal mobility of capturing devices have fueled an explosion of video and images, contributing to what we call Big Data [2], for the *most part*, well beyond the current bandwidth and real-time computing capabilities.

Interestingly, the emerging High-Efficiency Video Coding (HEVC) [3] standard for video compression, promises up to 50% bit rate savings compared against the best of compression schemes available today. While the compression efficiency of HEVC offers a unique opportunity to alleviate the bandwidth crunch, new methods of feature extraction directly from compressed video must be factored into the design of faster algorithms. However, most state-of-the-art (SoA) algorithms operate on uncompressed images with an independent background model for each pixel (pixel-based methods). Therefore, compressed videos have to undergo computationally intensive preprocessing to be completely de-compressed, claiming significant time and memory prior to the application of such algorithms. Furthermore, uncompressed

Manuscript received January 7, 2015; revised April 2, 2015; accepted May 31, 2015. Date of publication June 15, 2015; date of current version July 13, 2015. The work of M. K. Kundu was supported by the Indian National Academy of Engineering (INAE) through the INAE Distinguished Professor Fellowship. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yui-Lam Chan.

B. Dey is with the Center for Soft Computing Research, Indian Statistical Institute, Kolkata 700108, India (e-mail: bhaskar.dey09@gmail.com).

M. K. Kundu is with the Machine Intelligence Unit, Indian Statistical Institute, Kolkata 700108, India (e-mail: malay@isical.ac.in).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2015.2445631

image sequences contain a high degree of statistical redundancy. This is because neighboring pixels within an image (spatial redundancy) as well those between subsequent images (temporal redundancy) in video tend to have very similar intensities. Thus, pixel-based methods not only exhaust substantial resources for processing redundant data, they fail to exploit the correlation between adjacent pixels.

In this paper, a low-complexity technique for foreground segmentation is proposed. The method which relies on *spatio-temporal de-correlated block features* (ST-DBF) extracted from HEVC coded bitstream semantics. Real-time performances with accuracy comparable to those of pixel-based methods are targeted. The major contributions of the proposed method are:

1) A set of two features that sufficiently describe block coding units of HEVC compressed video is proposed. Video compression, in general, entails transformation of statistically correlated (or redundant) visual signals into highly de-correlated bitstream. As a matter of fact, the lesser the correlation between input signals, the higher is the compression achieved. The new HEVC coding standard convincingly outperforms its predecessors, i.e., H.264/AVC, MPEG-4 Visual, etc. in terms of compression efficiency. Consequently, HEVC coded video provides the best source of de-correlated data; the features derived from which enables faster machine learning using significantly fewer samples for training. It's important to mention here, that unlike the proposed ST-DBF, uncompressed natural images are typically high-dimensional data containing mostly redundant features. This pose severe computational challenges for pixel-based methods while learning and adapting to background models in real-time;
2) Unlike most pixel-based methods that require significant storage and computational overhead for decoding a compressed video prior to initialization, we propose its efficient *reuse* for the development of a block-based background model.
3) Offline or batch processing for initialization and update of background model parameters are completely replaced with an online process by using proposed recursive formulations.

The subsequent sections are organized as follows. Related work is presented in Section II. Section III introduces the proposed method. Experimental results are presented in Section IV, while Section V concludes the paper.

II. Related Methods

We summarize related work in this section; some of these are referred to in more detail depending on the context. Given the vast amount of literature in the field, it is not possible for us to provide a comprehensive survey, but we attempt to include the major trends. We refer the reader to several recent surveys [1], [4] for more details. In general, most background modeling methods strictly employ a model independently for each pixel-location. Early approaches surmised that the intensity of a pixel observed over time in a static scene could be modeled with a temporal median filter [5] and a single Gaussian distribution [6]. Over the years, increasingly complex algorithms have been proposed. Among these, by far the most popular is the Gaussian Mixture Model (GMM) presented by Friedman and Russel [7], and independently by Stauffer and Grimson [8]. This process consists in fitting the distribution of intensity values observed over time at each pixel-location by a weighted mixture of Gaussians. This model can cope with the multi-modal nature of many practical situations and leads to decent results when a repetitive background motion, such as waving trees, water ripples, etc. are encountered. The popularity of GMM led to many significant improvements, which include adaptively changing the number of Gaussians per pixel [9], [10], Conditional Random Field based GMM [11], and a multi-resolution approach [12]. More recently, a new method called Dirichlet process-based GMM (DPGMM) [13] has been proposed, which can determine the optimal number of mixture of components using a Bayesian formulation.

The non-parametric kernel density estimation (KDE)-based technique [14] is another popular approach. Unlike parametric fitting of a finite mixture of Gaussians, KDE is a more general approach that does not assume any specific shape for the underlying distribution. Reddy *et al.* [15] proposed block descriptors using the discrete cosine transform (DCT), which are classified into foreground or background using the cascaded output of three different classifiers. A robust approach for background subtraction on H.264/AVC compressed video was proposed in [16]. Other notable methods include a self-organizing approach to background subtraction (SOBS) [17], Robust Principal Component Analysis (RPCA) [18], compressed sensing [19], advanced fuzzy aggregation based background subtraction (AFABS) [20], and local variation persistence (LVP) [35].

Being the most bandwidth-efficient codec today, HEVC will likely be a very popular choice for video content delivery in the coming decade. In applications where minimizing bandwidth is not the highest priority, HEVC can still be used to significantly improve video quality at the same bitrate as H.264/AVC. With notable contributions [21]–[26] having made for further reducing the bandwidth required for storage/transmission of surveillance video, lots of media applications and products are currently pursuing the HEVC support. In literature, until recently there were no well-known algorithms for video content interpretation and analysis using features derived specifically from HEVC coded video. Off late, a novel research effort toward derivation of visual content using HEVC bitstream semantics [27] has been reported.

In this paper, we propose a fast and yet robust background subtraction approach for surveillance videos compressed in the HEVC format. The method exploits coded bitstream semantics to unlock object motion/activity patterns. A widespread commercial deployment of HEVC being clearly imminent, the proposed method brings in a tremendous potential to harness the computational power of integrated capturing and encoding devices.

III. Proposed Method

We present some contextual preliminaries on HEVC video compression and associated picture partitioning syntax based on which block features are derived. Interested readers may refer to a comprehensive overview [3] for more details.

A. Preliminaries

Video signals in HEVC typically use YCbCr color space with 4:2:0 sampling. This separates a color representation into three components called Y, Cb, and Cr. The Y component is also called luma, and represents bright-ness. The two chroma components Cb and Cr represent the extent to which the color deviates from gray toward blue and red, respectively. Since the human visual system is more sensitive to brightness than color, the 4:2:0 sampling structure is used, in which each chroma component has only one-fourth of the number of samples of the luma component (half the number of samples in both the horizontal and vertical dimensions). The sparse sampling of the chroma components, which enables relatively less data to be coded without noticeable difference, is attributed to the phenomena known as *psychovisual* redundancy. The feasibility of video compression, however, rests mainly with two other types of redundancies, i.e., *temporal* and *spatial* redundancy. While the former results from individual frames being highly correlated with neighboring frames, the latter is due to similarity of adjacent pixels within a single frame.

A compressed HEVC video consists of a sequence of pictures or frames, each of which is split into non-overlapping blocks called coding tree unit (CTU). The CTU is the fundamental unit of compression, which maintains information for each color component in structures known as the coded tree block (CTB). A luma CTB covers a rectangular picture area of $L \times L$ samples of the luma component and the corresponding chroma CTBs cover $L/2 \times L/2$ samples of each of the two chroma components. The value of L (heretofore used to denote the size of luma CTBs) for a given sequence is fixed by the encoder and signaled by a sequence parameter which may be either equal to 16, 32, or 64. Each CTB can be split recursively into a quadtree structure, all the way down to 8×8 (in units of luma samples) regions. The quadtree structure is known as the coding quadtree (CQT). So, for the example shown in Fig. 1 (top), the 64×64 luma CTB corresponding to the CTU at location 14 is shown to consist of two 32×32, six 16×16, and eight 8×8 regions. These regions are called coding blocks (CBs). The spatial and the temporal redundancies respectively of a given CB are reduced by splitting it into blocks that were predicted from previously coded blocks within the same frame (called intra-prediction), as well as from the neighboring frames (called inter-prediction). The blocks are called prediction blocks (PBs). Inter-prediction of a PB is a temporal de-correlation technique by which one or two suitable reference blocks are selected. The reference blocks are indicated with offsets relative to the current block, in both horizontal and vertical directions. The prediction information of a PB is indicated as a motion vector (MV) corresponding to the offsets, and a reference frame index pointing to the referenced frame. The error in intra/inter-prediction of a CB,

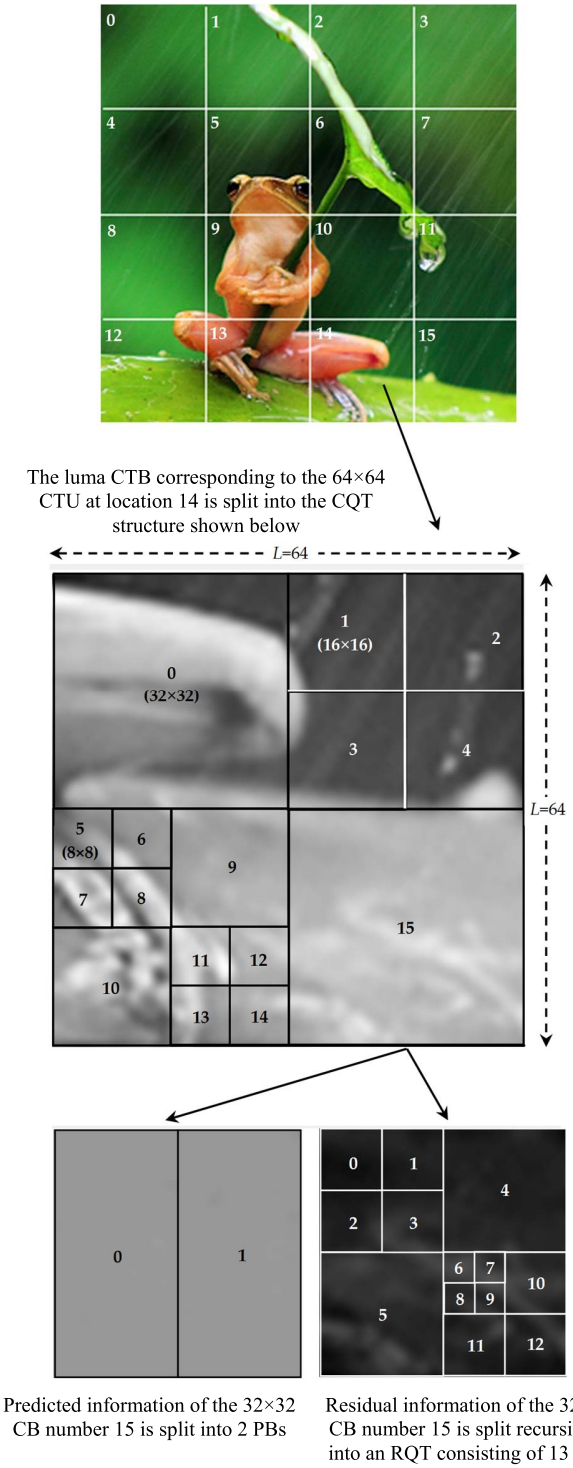

Fig. 1. An example of a 256×256 image (not to scale) partitioned into sixteen 64×64 CTUs (top image). The luma CTB corresponding to CTU at location 14 is split into a CQT structure (middle image). The CB number 15 of the CQT is shown to be coded using a predicted information consisting of 2 PBs (shown just above, to the left) and the prediction residual, which is split into an RQT consisting of 13 TBs (shown just above, to the right).

i.e., the residual, still contains correlation between spatially neighboring pixels. Therefore, the residual is further split into a quadtree structure, called the residual quadtree (RQT), into transform blocks of suitable sizes ranging from 4×4 to 32×32 samples. The TBs are subjected to transform coding and quantization. The purpose of transform coding is

to decompose a batch of correlated signal samples into a set of uncorrelated spectral coefficients, with energy concentrated in as few coefficients as possible. This compaction of energy permits a prioritization of the coefficients, with the more energetic ones receiving a greater allocation of encoding bits. The resulting transform coefficients are quantized. The purpose of quantization is to map a large set of input coefficients to a smaller set of representative quantization levels. This results in rounding of coefficient values to some acceptable unit of precision specified by a quantization parameter (*QP*). The output of a quantizer is typically a sparse array of quantized coefficient levels, mainly containing zeros. Finally, the non-zero coefficient levels are entropy coded. This process eliminates *coding* redundancy by assigning fewer bits to more frequently-occurring symbols.

B. The CTU Features

In HEVC compressed video, MVs and transform coefficients of a CTU registers only incremental changes occurring between adjacent frames. Major changes correspond to moving objects, while others are due to non-stationary elements of the background. It was verified, using a video codec analyzer,[1] that the CTUs enclosing parts of moving objects contained MVs and transform coefficients of significantly higher energy than those *normally* corresponding to the scene background. Hence, the energy associated with the prediction information (i.e., MVs, reference indices) and the prediction error/residual (i.e., transform coefficient levels) of a CTU are adopted as indicators or features of a potential foreground activity/motion in a coded video sequence. The features associated with the prediction and the residual are, hereafter, denoted by non-negative random variables y and x respectively.

Without loss of generality, we assume C CTUs per encoded picture or frame. CTUs in a given picture are addressed/indexed numerically using a location parameter $idx \in \{0, 1, \ldots, C-1\}$ in the *raster-scan* order starting with $idx = 0$ for the CTU at the top-left-hand corner. An example is shown in Fig. 1 (top), where a 256×256 sized picture is aggregated into a set of sixteen CTUs. Specifically, the features corresponding to the CTU located at *idx* in frame t are denoted as the 2×1 pattern vector

$$\vec{F}_{t,idx} = \left(x_{t,idx}, \quad y_{t,idx}\right)^T,$$

where $x_{t,idx}$, $y_{t,idx}$ are particular instances of the variables x and y respectively. Computation of $x_{t,idx}$ and $y_{t,idx}$ are described in the following sections.

C. Computation of CTU Feature $x_{t,idx}$

We describe $x_{t,idx}$ as the energy associated with residual transform coefficients the CTU located at *idx* in frame t. As discussed earlier, the residuals obtained by subtracting the predicted pixels from those of the target image blocks are subjected to transform coding and quantization. HEVC specifies integer approximations of the DCT for transform sizes 32×32, 16×16, 8×8, and 4×4 [28]. Alternatively, the discrete sine transform (DST) is used only for 4×4 intra-predicted luma TBs. In the following, we propose a statistical formulation of $x_{t,idx}$ based on the following CB information extracted from the RQT: 1) the quantization parameter *QP*; and 2) the number of bits B consumed in coding the residual transform coefficients for each TB.

The 2D auto-regressive models of the first order, i.e., 2D AR (1), have extensively been used to represent natural images owing to its capability to provide robust estimations for the intensity of pixels by a small number of parameters. The residues practically have a zero mean value; hence, we model a residual TB as *stationary* 2D AR(1) signal source X of the form

$$\begin{aligned} X(i,j) = {} & \rho_h X(i-1,j) + \rho_v X(i,j-1) \\ & -\rho_h \rho_v X(i-1,j-1) + \xi(i,j), \end{aligned} \tag{1}$$

where ξ is a zero-mean white noise with unit variance, and ρ_{h}, ρ_{v} denote the first-order horizontal and vertical correlation coefficients respectively. Under stationary conditions, the 2D correlation function is *separable* and may be expressed as [29]

$$R(i,j) = \sigma_X^2 \rho_h^{|i|} \rho_v^{|j|}, \tag{2}$$

where

$$\sigma_X^2 = 1/[(1-\rho_h^2)(1-\rho_v^2)].$$

Considering a generic $N \times N$ residual TB, the resultant block of transformed coefficients Z may be specified as a matrix product

$$Z = AXA^T, \tag{3}$$

where A is the $N \times N$ orthogonal transformation basis matrix. For DCT (Type-2), the $(i,j)^{\mathrm{th}}$ element of A is

$$A(i,j) = \begin{cases} 1/\sqrt{N} & i = 0, \quad 0 \le j < N \\ \sqrt{2/N}\cos\left(i\pi(2j+1)/2N\right) & 1 \le i < N, \quad 0 \le j < N, \end{cases} \tag{4}$$

while the same for DST (Type-7) being

$$A(i,j) = \frac{2}{\sqrt{2N+1}} \sin\frac{(2i+1)(j+1)\pi}{2N+1} \quad 0 \le i, \quad j < N. \tag{5}$$

The statistical distributions of transform coefficients are best modelled in literature [30] as a zero mean Laplacian probability density function (pdf), i.e.,

$$f(z) = \frac{1}{2\lambda}\exp\left(-|z|/\lambda\right), \quad z \in \mathbb{R} \tag{6}$$

where z is the coefficient value of a particular frequency component, and $\lambda > 0$ being a parameter that determines the coefficient variance, i.e., $2\lambda^2$. Under conditions (2), the coefficient variance of the $(u,v)^{\mathrm{th}}$ component of a TB, say $\sigma_Z^2(u,v)$, may be given as [31]

$$\sigma_Z^2(u,v) = \sigma_X^2 \left[ARA^T\right]_{u,u} \left[ARA^T\right]_{v,v}, \tag{7}$$

where $[\cdot]_{u,u}$ is the $(u,u)^{\mathrm{th}}$ element of the argument matrix. Therefore, we express the probability distribution of the

[1]Elecard HEVC Analyzer: www.elecard.com

coefficients of Z as a weighted mixture of N^2 Laplacian densities (6), each corresponding to individual frequency component as

$$f_{TB}^{N}(z) = \frac{1}{\sqrt{2}N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \frac{1}{\sigma_Z(u,v)} \exp\left(-\frac{\sqrt{2}}{\sigma_Z(u,v)}\,|z|\right). \tag{8}$$

As discussed earlier, the transform coefficients are quantized, with the degree of quantization signaled by the RQT using $QP \in \{0, 1, 2, \ldots, 51\}$. In general terms, quantization of an input coefficient z is performed according to

$$k = \text{round}(z/Q) = \text{sgn}(z)\,\lfloor(|z| + f)/Q\rfloor, \tag{9}$$

where $k \in \{0, \pm1, \pm2, \ldots\}$ is the mapped *quantization level* for all values of z in the corresponding quantization interval, Q represents the width of quantization intervals determined by N, QP as [28]

$$Q = 2^{21+\lfloor QP/6\rfloor - \log_2 N}, \tag{10}$$

and f is a rounding offset parameter equal to $Q/3$ and $Q/6$ for the intra-predicted and the inter-predicted CBs respectively. From (9), it is easy to check that all values of z in the interval $(-Q+f, Q-f)$ are mapped to $k = 0$. Similarly, for $k = -1, -2, -3, \ldots$ the respective quantization intervals are $((kQ - Q + f), (kQ + f)]$, while those for $k = 1, 2, 3, \ldots$ being equal to $[(kQ - f), (kQ + Q - f))$. Therefore, the probability $P(z_k)$ that a random input coefficient of Z is mapped to level k, may be analytically expressed as the definite integral of (8) over the k^{th} quantization interval

$$P(z_k) = \begin{cases} \displaystyle\int_{(kQ-Q+f)}^{(kQ+f)} f_{TB}^{N}(z)dz \\ \displaystyle= \frac{1}{2N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \exp\left(\frac{\sqrt{2}(kQ+f)}{\sigma_Z(u,v)}\right) \times \left(1 - \exp\left(-\frac{\sqrt{2}Q}{\sigma_Z(u,v)}\right)\right) & k < 0 \\ \displaystyle\int_{(-Q+f)}^{(Q-f)} f_{TB}^{N}(z)dz & \\ \displaystyle= \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} (1 - \exp(-U)) & k = 0 \\ \displaystyle\int_{(kQ-f)}^{(kQ+Q-f)} f_{TB}^{N}(z)dz & \\ \displaystyle= \frac{1}{2N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \exp\left(\frac{\sqrt{2}(-kQ+f)}{\sigma_Z(u,v)}\right) \times \left(1 - \exp\left(-\frac{\sqrt{2}Q}{\sigma_Z(u,v)}\right)\right) & k > 0, \end{cases} \tag{11}$$

where

$$U = \sqrt{2}(Q - f)/\sigma_Z(u, v).$$

Using the level information k, the decoding process simply reconstructs the quantized coefficients z_k as

$$z_k = kQ. \tag{12}$$

Please note that a given k, the computation process of z_k and $P(z_k)$ requires only N and QP (which is related to Q be known. Using (11) and (12), the energy x_{TB} associated with the coefficients of an $N \times N$ TB may be given as

$$x_{TB}(N, QP) = \sum_{k=-\infty}^{\infty} z_k^2 P(z_k) = 2\sum_{k=1}^{\infty} z_k^2 P(z_k),$$

which on simplification yields

$$x_{TB}(N, QP) = \frac{Q^2}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \frac{\left(1 + \exp\left(-\frac{\sqrt{2}Q}{\sigma_Z(u,v)}\right)\right)\exp(-U)}{\left(1 - \exp\left(-\frac{\sqrt{2}Q}{\sigma_Z(u,v)}\right)\right)^2}. \tag{13}$$

While the values of N and QP directly from the RQT, $\sigma_Z(u, v)$ is indirectly dependent on ρ_{h}, ρ_{v} via (7) and (2). We assume, without loss of generality, that the residual inter-pixel correlations are the same in both horizontal and vertical directions, i.e., $\rho_{\text{h}} = \rho_{\text{v}} = \rho$ (say). Therefore, the value of ρ is estimated by equating an expression for the number of bits B consumed in coding the residual TB with its actual value obtained from the RQT during decoding.

We formulate an expression of B as follows. Recall that the lower bound on the *average* number of bits required to encode a quantized coefficient may be given by the (discrete) entropy measure

$$H(Z) = -\sum_{k=-\infty}^{\infty} P(z_k)\log_2(P(z_k)). \tag{14}$$

The entropy of a mixture density, in general, cannot be obtained in a closed form due to its inherent difficulty in evaluating the logarithm of a sum of exponential functions. However, we observe that the entropy $H(Z)$ is a *concave function* of the probability distribution of a randomly picked coefficient of Z, which is considered to be a mixture of N^2 Laplacian distributed variables of equal weightage, say $z_{0,0}, z_{0,1}, \ldots, z_{N-1,N-1}$, i.e.,

$$Z = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \frac{1}{N^2} z_{u,v}.$$

Therefore, by Jensen's inequality, we have

$$H(Z) = H\left(\sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \frac{1}{N^2} z_{u,v}\right) \geq \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \frac{1}{N^2} H(z_{u,v}) = H_{approx}(Z),$$

which is a lower bound on $H(Z)$. Simplification of $H_{approx}(Z)$ yields

$$H_{approx}(Z) = \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} H(z_{u,v}), \tag{15}$$

where

$$H\left(z_{u,v}\right) = -\left(1-\exp\left(-U\right)\right)\log_2\left(1-\exp\left(-U\right)\right) - \frac{\sqrt{2}\exp\left(-U\right)}{\sigma_Z(u,v)\ln 2}\left(f - \frac{Q}{1-\exp\left(-\frac{\sqrt{2}Q)}{\sigma_Z(u,v)}\right)}\right) - \exp\left(-U\right)\left(\log_2\left(1-\exp\left(-\frac{\sqrt{2}Q}{\sigma_Z(u,v)}\right)\right)-1\right).$$

The expression for the number of transform coding bits B for an $N \times N$ TB is given by

$$B = N^2 H_{approx}(Z) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} H(z_{u,v}). \tag{16}$$

Substituting the values of B, Q, and f in (16), the unknown parameter ρ unknown parameter is evaluated using numerical methods. As discussed earlier, the value of ρ is used to compute $\sigma_Z(u,v)$, which in turn is substituted in (13) to evaluate x_{TB}. Given the fact that B and QP assume only discrete non-negative values from a limited range, we enumerate all possible combinations of (B, QP) for all $N \in \{4, 8, 16, 32\}$ and construct lookup tables (LUTs) containing precomputed values of x_{TB}. The use of LUTs, therefore, replaces the complexity of repetitive numerical computations at run-time with only a few lookup operations.

We assume p TBs of sizes $N_1 \times N_1$, $N_2 \times N_2, \ldots, N_{\mathrm{p}} \times N_{\mathrm{p}}$ comprising a given CB. Therefore, the energy x_{CB} of the CB with known QP (which is fixed for all constituent TBs) and size $M \times M$ may be expressed as the weighted sum of the energy associated with individual TBs, i.e.,

$$x_{CB}(M, QP) = \frac{1}{M^2}\sum_{i=1}^{p} N_i^2 x_{TB}(N_i, QP). \tag{17}$$

A normalization factor or weight (N_i^2/M^2) is associated with the i^{th} TB in proportion to the CB area it represents. Finally, assuming q CBs of sizes $M_1 \times M_1, M_2 \times M_2, \ldots, M_{\mathrm{q}} \times M_{\mathrm{q}}$ and respective quantization parameters $QP_1, QP_2, \ldots, QP_q$ comprising a given CTU, we have

$$x_{t,idx} = \frac{1}{L^2}\sum_{i=1}^{q} r_i M_i^2 x_{CB}(M_i, QP_i), \tag{18}$$

where r_i is a multiplier equal to 2/3 or 1/6 accordingly as pixels of the i^{th} CB are of luma (Y) or chroma components (Cb or Cr) respectively. The multiplier ensures that the weights sum up to unity, i.e., $\sum_{i=1}^{q}\left(r_i M_i^2/L^2\right) = 1$.

It may be noted that the proposed formulation of $x_{t,idx}$ is based on the transform domain statistics using only coded information such as B, QP, N, f for each PB/TB rather than actual coefficients decoded from a compressed video. Considering the fact that discrete trigonometric transforms output exactly the same number of coefficients as input pixels, the statistical formulation (18) helps us evaluate $x_{t,idx}$ without involving a complexity equivalent to those of the pixel-based methods. In fact, the proposed formulation of F_1 replaces all run-time computations with a few lookup operations. In Section IV-A, we compare the predicted values of $x_{t,idx}$ with those that are computed directly from the decoded coefficients.

D. Computation of CTU Feature $y_{t,idx}$

Unlike $x_{t,idx}$ which quantifies the energy associated with residual or prediction error, $y_{t,idx}$ is the energy associated with CTU prediction. The prediction mode is signaled at the CB-level as being intra or inter. Each luma and chroma CBs constituting a CTU is split into one, two, or four PBs. It may be noted that the PBs are essentially rectangular (instead of being a square) in the case when a CB is split into two partitions. The intra-prediction mode of a CB implies that the constituent PBs were predicted from previously coded samples from one (uni-directional, i.e., either forward or backward) or two (bi-directional – both forward as well as backward) reference frames.

It may be noted that the MV magnitudes corresponding to a given PB encode the predicted part of local incremental changes occurring in a scene. Therefore, the prediction signal energy of a CB is computed as the weighted sum of the squared MV magnitudes of all constituent PBs, where the weights represent the portion (in terms of sample size) of the CB accounted for by each PB. Formally, we denote $(h_{i,j}, v_{i,j})$ as the MV corresponding to the i^{th} PB in the j^{th} predicted direction (either forward or backward), where $h_{i,j}$ and $v_{i,j}$ represent the offsets in the horizontal and the vertical directions respectively. Let $s_{i,j}$ be the reference frame index corresponding to $(h_{i,j}, v_{i,j})$. The value of $s_{i,j}$ implies that the current frame and the frame being referenced are at a temporal distance of $(s_{i,j}+1)$, between which a change due to motion/activity must have occurred. Hence, the individual energy term associated with the squared MV magnitudes, i.e., $(h_{i,j}^2 + v_{i,j}^2)$ is normalized by$(s_{i,j}+1)$. Furthermore, let d_i be equal to 1 or 2 accordingly as the i^{th} PB is uni-directionally or bi-directionally predicted. We assume a total of p PBs of rectangular sizes $W_1 \times H_1, W_2 \times H_2, \ldots, W_{\mathrm{p}} \times H_{\mathrm{p}}$ constituting a given $M \times M$ CB. Therefore, the energy y_{CB} associated with the CB is given by

$$y_{CB}(M) = \frac{1}{M^2}\sum_{i=1}^{p}\frac{W_i H_i}{d_i}\sum_{j=1}^{d_i}\left((h_{i,j}^2 + v_{i,j}^2)/(s_{i,j}+1)\right), \tag{19}$$

where weights $(W_i H_i/M^2)$ and $(1/d_i)$ are respectively due to the sample size and the number of MVs associated with the i^{th} PB.

Finally, assuming q CBs of sizes $M_1 \times M_1$, $M_2 \times M_2, \ldots, M_{\mathrm{q}} \times M_{\mathrm{q}}$ comprising the luma CTB (since the same prediction information used for the luma components is used by the chroma components as well) of a given CTU, we have

$$y_{t,idx} = \frac{1}{L^2}\sum_{i=1}^{q} M_i^2 y_{CB}(M_i). \tag{20}$$

It may be noted that the minimum PB size is 8×4 or 4×8 for unidirectional prediction and 8×8 for bi-directional prediction, which accounts for a maximum of 128 MVs for

a 64×64 CTU. Additionally, in (19), a maximum of five multiplications / divisions and two additions is required for each MV. Therefore, computation of $y_{t.idx}$ requires no more than 640 multiplications / divisions and 256 additions. The information related to PB sizes, MVs, and reference indices that are required to compute $y_{t.idx}$ is parsed from the CQT.

E. Interpretation of the CTU Pattern

The proposed CTU pattern $\vec{F}_{t,idx}$ quantify only incremental changes occurring at a particular region (defined by the CTU at location *idx*) between successive frames of a sequence. Since video is a representation of continuous events in time, a substantial change at any given time is indicative of potential foreground activity. Specifically, the temporal history of $\vec{F}_{t,idx}$ observed at location *idx*, provides $|\Sigma_{idx}|$ - the magnitude of covariance between variables x and y as a parameter that is consistent with recent activity/changes occurring at that location. The distribution/scatter of $\vec{F}_{t.idx}$ for stationary and non-stationary locations of various sequences coded in HEVC were noted. Fig. 2 shows a frame of a typical dynamic sequence with a waving tree in the background. Grids (in light-gray) are overlaid on the frame to indicate the CTU demarcations. The scatter-plots of $\vec{F}_{t,idx}$ observed for $t = 1, 2, \ldots, 200$ corresponding to locations at $idx = \{589, 706\}$ are shown as indicated by arrows. The location at 706 depicts static background. As shown in the corresponding scatter-plot, this is modeled effectively with a highly dense cluster with $|\Sigma_{706}| = 0.863$. The same background is also seen to portray dynamic entities (i.e., waving tree branches) at location 589, which is characterized by a widely distributed cluster with a significantly higher value of $|\Sigma_{589}| = 2.034$. In both cases, the proposed CTU features are found to be practically uncorrelated as the correlation coefficients is close to zero for both locations.

F. Background Model Development

As highlighted in the introductory section, the proposed foreground segmentation method involve binary decisions at *two* levels of granularity as follows:

1. Performing a coarse block-level segmentation of each frame by selecting of a set of potential CTUs that are occupied fully or partially by parts of moving objects;
2. Performing a finer pixel-level segmentation by eliminating pixels from the selected CTUs that are similar (in intensity) to the corresponding background model.

Therefore, our coarse-to-fine approach requires background models to be initialized both at the CTU and the pixel level as follows. The set of initial T (say, $T = 200$) frames of a sequence are used for unsupervised training of background model parameters. For pixel-based model initialization, the temporal median of a set of frames selected at regular intervals is computed. Although the initial sequence is *ideally* supposed to contain only background information, in practice however, it is difficult to get real-world sequences devoid of foreground appearances. It may be noted that the median has a breakdown value 1/2, i.e., it can tolerate upto 50% outliers. Therefore, the median is chosen as a robust statistic that proves useful in

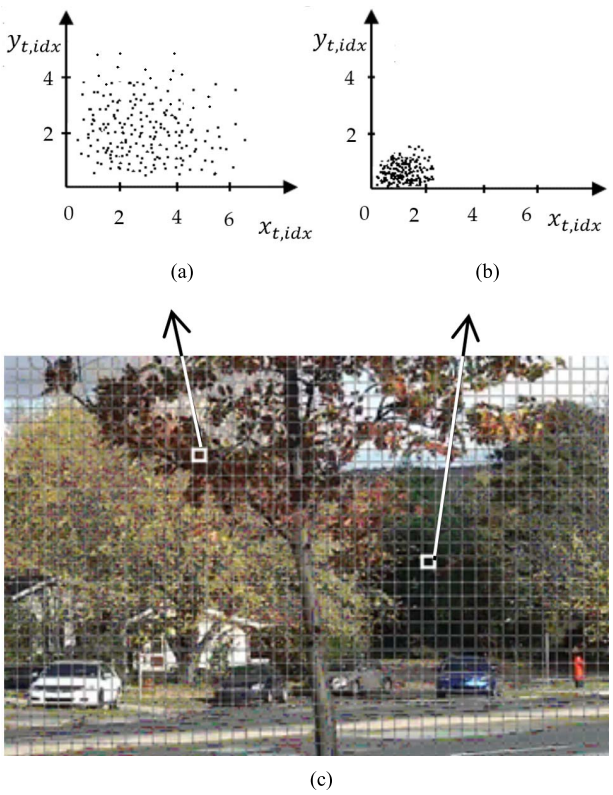

Fig. 2. Interpretation of the proposed CTU features. (a) Cluster plot for CTU #589 Correlation coefficient = 0.0806. (b) Cluster plot for CTU #706 Correlation coefficient = 0.071. (c) Frame #200 of the *Fall* sequence depicting a waving tree at the background. Square blocks corresponding to the CTU locations numbered 589 and 706 are highlighted in white.

filtering out foreground objects appearing in the initial frames. Simultaneously, a block-based background model is initialized by computing the mean $\vec{\mu}_{idx}$ and co-variance Σ_{idx} for the cluster of CTU patterns $\vec{F}_{t,idx}$ observed over the training history $t = \{1, 2, \ldots, T\}$. The parameters $\vec{\mu}_{idx}$ and Σ_{idx} are computed for all $idx \in \{0, 1, \ldots, C-1\}$, where C is the number of clusters or CTUs per frame. Computation of $\vec{\mu}_{idx}$ and Σ_{idx} is implemented on-line via a recursive process (described later in Section III-G), which enables us to update the model parameters with only a few additions/multiplications for every incoming pattern.

Following the training phase, the first natural step in segmenting a new frame (with $t > T$) is to compare CTU patterns for similarity with the corresponding background cluster. Similarity is ascertained if the Mahalanobis distance D measured between the incoming pattern $\vec{F}_{t,idx}$ and its corresponding cluster (background) centroid $\vec{\mu}_{idx}$ falls below a desired threshold α. For a given frame t, let S_t be the subset of CTUs satisfying

$$D = \sqrt{\left(\vec{F}_{t,idx} - \vec{\mu}_{idx}\right)^T \Sigma_{idx}^{-1} \left(\vec{F}_{t,idx} - \vec{\mu}_{idx}\right)} > \alpha, \quad (21)$$

for all $idx \in \{0, 1, \ldots, C-1\}$. The value of α is adaptively selected, as discussed later in the section. The CTUs in S_t, comprise a coarse *segmented region* of the foreground in frame t. Since real object boundaries rarely follow block boundaries, pixel-level refinement of the CTUs $\in S_t$ is

performed by eliminating pixels from the segmented region that are similar to the co-located pixels in the background frame. Considering a given pair of pixels with YCbCr color coordinates $X_{\mathrm{F}} \equiv (Y_{\mathrm{F}}, Cb_{\mathrm{F}}, Cr_{\mathrm{F}})$ and $\mathrm{X_B} \equiv (Y_{\mathrm{B}}, Cb_{\mathrm{B}}, Cr_{\mathrm{B}})$ respectively from the segmented region and the background frame, let luminance differential $\Delta \mathrm{Y} = |\mathrm{Y}_F - \mathrm{Y}_B|$ and chrominance differential $\Delta C = |\mathrm{Cb}_F - \mathrm{Cb}_B| + |\mathrm{Cr}_F - \mathrm{Cr}_B|$. $\mathrm{X_F}$ is classified as foreground if $(\Delta Y/|\Sigma_{idx}|) > t_Y$ and $(\Delta C/|\Sigma_{idx}|) > t_C$, where t_{Y} and t_{C} are decision thresholds determined empirically as 0.05 and 0.03 respectively.

For a given CTU $\in S_t$, the number of pixels which correspond to the foreground say m, determines the value of adaptive threshold α. It is initially chosen as the value corresponding to 99% *prediction interval* for background clusters. As *a priori* information about the specific shape of each cluster is not available, we assume the *normally* distributed case for all $idx \in \{0, 1, \ldots, C-1\}$. Therefore, letting $\vec{\mu}_{idx} = (\mu_x, \mu_y)^T$ and $\Sigma_{idx} = \mathrm{Diag}(\sigma_x^2, \sigma_y^2)$, α is obtained by equating the estimated background probability $P_B = 0.99$ with its analytical expression obtained over the 2D interval $\left[\left(\mu_x - \alpha\sigma_x, \mu_y - \alpha\sigma_y\right), \left(\mu_x + \alpha\sigma_x, \mu_y + \alpha\sigma_y\right)\right]$, i.e.,

$$P_B = \int_{\mu_y-\alpha\sigma_y}^{\mu_y+\alpha\sigma_y} \int_{\mu_x-\alpha\sigma_x}^{\mu_x+\alpha\sigma_x} \frac{1}{2\pi\sigma_x\sigma_y} \exp\left[-\left(\frac{(x-\mu_x)^2}{2\sigma_x^2} + \frac{(y-\mu_y)^2}{2\sigma_y^2}\right)\right] dxdy = \mathrm{erf}^2\left(\frac{\alpha}{\sqrt{2}}\right) = 0.99 \Rightarrow \alpha \approx 2.8. \quad (22)$$

It is observed, that if a CTU location in the current frame is occupied (partially or fully) by parts of a foreground object, then the value of P_B for the collocated CTU patterns decreases abruptly in *subsequent* frames. Therefore, α is adapted based on the value of m in the current frame. For an $L \times L$ CTU, we ideally have $P_B = [1 - (m/L^2)]$, which is used to compute the value of α for the collocated CTU in the next frame as

$$\alpha = \sqrt{2}\mathrm{erf}^{-1}\left(\sqrt{P_B}\right). \quad (23)$$

Theoretically, subsequent CTU patterns for $m = L^2$ will never be detected as foreground, while for $m = 0$ any CTU pattern will be considered as foreground. This will lead to a deadlock situation with the CTU patterns being continuously detected as either foreground or background. Therefore, for all practical cases $m \in \{0, 1, \ldots, L^2\}$, we use the approximation $P_B \approx [0.99 - (0.98m/L^2)]$ such that P_B is always restricted to the values in [0.01, 0.99]. The adaptive thresholding process is found to be effective in detecting objects that move with a higher variation of speed in the field of view and those with intermittent motion.

G. Recursive Computation of $\vec{\mu}_{idx}$ and Σ_{idx}

In order to compute/update $\vec{\mu}_{idx}$ and Σ_{idx} on-the-fly, we consider a temporal sliding window containing a *maximum* of K *most recent* samples of $\vec{F}_{t,idx}$ from the history of location *idx*. With every incoming frame t, a new CTU pattern enters the window. If $t > K$, this causes the least recent pattern $\vec{F}_{t-K,idx}$ to first exit the window (in first-in-first-out manner) to make room for the new pattern. Therefore, after sliding past n frames of a sequence, the window remains populated with a sequence of $\mathrm{V}_n = \min(K, n)$ CTU patterns given by $\{(x_{t,idx}, y_{t,idx})^T\}_{t=max(1,n-K+1)}^{n}$, which determine the current values of $\vec{\mu}_{idx}$ and Σ_{idx}. Given the current state n of the window, let $E_n[x^a y^b]$ denote the *expected value* of $(x^a y^b)$, where the ordered pair $(a, b) \in \{(1,0), (0,1), (2,0), (0,2)\}$. Using the expected value notation, $\vec{\mu}_{idx}$ and (bias corrected) Σ_{idx} is given by

$$[\vec{\mu}_{idx}]_n = (\, E_n[x], \quad E_n[y]\,)^T \quad (24)$$

and

$$[\Sigma_{idx}]_n = \frac{V_n}{V_n - 1}\begin{pmatrix} E_n[x^2] - (E_n[x])^2 & 0 \\ 0 & E_n[y^2] - (E_n[y])^2 \end{pmatrix} \quad (25)$$

respectively. The off-diagonal terms in (24), which indicate the covariance between practically uncorrelated variables x and y, are taken as zero. The n^{th} update step of $E_n[x^a y^b]$ is defined by the recursion

$$E_n[x^a y^b] = \begin{cases} x_{1,idx}^a y_{1,idx}^b & \text{for } n = 1, \\ \frac{1}{n}\left((n-1)E_{n-1}[x^a y^b] + x_{n,idx}^a y_{n,idx}^b\right) & \text{for } 1 < n \le K, \\ E_{n-1}[x^a y^b] + \frac{1}{K}(x_{n,idx}^a y_{n,idx}^b - x_{n-K,idx}^a y_{n-K,idx}^b) & \text{for } n > K. \end{cases} \quad (26)$$

It may be noted that the overall update of $\vec{\mu}_{idx}$ and Σ_{idx} using (23), (24) requires no more than 11 additions (or subtractions) and 15 multiplications (or divisions). The value of Σ_{idx} obtained as described above is susceptible outliers, i.e., foreground regions appearing in initial frames. Therefore, instead of using the pattern $(x_{t,idx}, y_{t,idx})^T$ directly for computing Σ_{idx}, we use the temporal medoid of $(x_{t,idx}, y_{t,idx})^T$, $(x_{t-1,idx}, y_{t-1,idx})^T$, and $(x_{t-2,idx}, y_{t-2,idx})^T$ as its robust counterpart. This helps obtain a robust estimate of Σ_{idx} for a negligible overhead of six floating-point comparisons in the worst case.

IV. Experimental Results and Discussion

This section is divided into three sub-sections. We first provide an experimental validation of the statistically predicted values of CTU feature x against its respective counterparts computed directly from the decoded bitstream. Secondly, we provide qualitative as well as quantitative comparison of the proposed method with those of the SoA on standard datasets. Finally, the impact of the size of CTU blocks on performance is discussed.

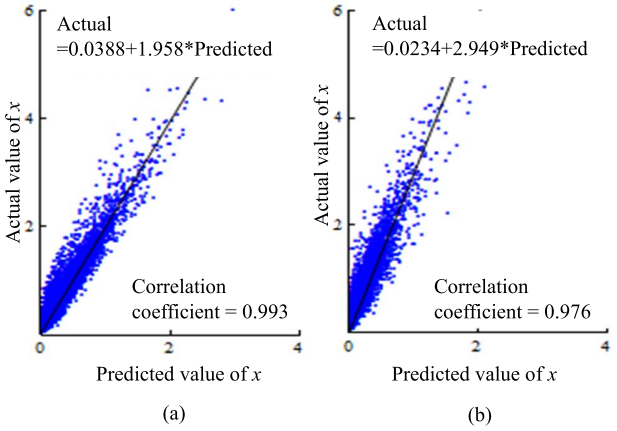

Fig. 3. Comparison of the predicted and the actual values of the proposed CTU feature x. Regression lines are obtained by the least square method. (a) idx = 706 (static location). (b) idx = 589 (dynamic location).

A. Validation of the Predicted Value of CTU Feature x

In Section III-C, we presented a statistical formulation of the CTU residual energy x based on the information parsed from RQT. The predicted values of x are validated experimentally against their actual counterparts computed from the decoded coefficients. Fig. 3 illustrates the comparison of the predicted and the actual values of $\{x_{t,idx}\}_{t=1}^{4000}$; $idx \in \{589, 706\}$ plotted on x-axis and y-axis respectively for the *Fall* sequence. The original (uncompressed) sequence was encoded in HEVC at a constant bit-rate of 500 KB/s. It is observed that the magnitudes of the predicted values are slightly lower compared to those computed directly from the decoded coefficients. This comes as a direct consequence of the fact that prediction of $x_{t.idx}$ depends on B, whose *lower-bound* on the average bit-rate was modeled using the entropy measure. Furthermore, the predicted value and its actual counterpart are found to be linearly correlated; in fact, the correlation coefficient was found to be greater than 0.96 for all recorded cases. The discrepancy between the predicted and the actual values, however, do not affect segmentation decisions taken using (23), because the Mahalanobis distance is known to be invariant under arbitrary linear transformations of the feature space. Therefore, in order to support faster computation, the predicted value of x is used as a convenient and yet justified surrogate to its actual counterpart; as the latter would otherwise have involved prohibitively high complexity due to coefficient level decoding.

B. Qualitative and Quantitative Evaluation

Experiments were conducted on challenge data sets used for the Change Detection workshop (CD.net) [32] and the Background Models Challenge (BMC) [33]. The CD.net data set, unlike any other publicly available, is very challenging and comprehensive. It includes 31 real-world (non-synthetic) sequences captured in diverse environments. All sequences of the data set are accompanied by accurate ground-truth segmentation of change/motion areas for each video frame that are subject to evaluation. The BMC data, on the other hand, encapsulates both synthetic and real-world sequences along with encrypted ground-truth masks for selected frames.

All sequences were considered for evaluation, with the original sequences transcoded to HEVC format using FFmpeg [34]. The encoder configuration was set as follows: Main profile with YCbCr 4:2:0 sampling and CTU sizes 16×16, weighted prediction as applied for both uni-directional and bi-directional cases, rate control was enabled (at the CTU level) with a target bit rate of 512 MB/s for all sequences, and video frames streaming at 25 frames/second. It is important to mention here, that in most streaming video applications, a predetermined output bit rate is desired. These applications, referred to as constant bit rate (CBR) applications, use a rate control strategy ensure a target bit rate by carefully selecting a different QP for each CBs comprising a CTU. The proposed method was implemented in C (built on MingW 64-bit platform) as a software patch and integrated into the HEVC decoding module of FFmpeg.

For comparison, we chose a set of five SoA methods [13], [15]–[17], [20]. Fig. 4 and Fig. 5 visually compare the segmentation results of six sequences chosen randomly from CD.net and BMC data sets respectively. The ground-truth and segmented frames contain black, white, and grayed-out portions, which respectively annotate the background, the foreground, and the regions that do not belong to the ROIs (which are, therefore, not evaluated). Based on similarity with the provided ground-truth data, clearly our algorithm performs qualitatively as well or better than the other techniques.

In order to compare the achieved results quantitatively with those of the SoA methods, we consider the performance on both the data sets, as reported in Table-I. The metrics adopted for quantitative evaluation includes

$$\text{Recall (Re)} = \#\text{TP}/(\#\text{TP} + \#\text{FN}), \tag{27}$$

$$\text{Precision (Pr)} = \#\text{TP}/(\#\text{TP} + \#\text{FP}), \tag{28}$$

$$\text{F-measure} = \frac{2 \times \text{Pr} \times \text{Re}}{\text{Pr} + \text{Re}}, \tag{29}$$

and average processing speed (frames/sec). The notations #TP, #FP, and #FN are total number of true positives, false positives, and false negatives (in terms of pixels), respectively. The first three metrics are well established measures for accessing binary segmentation / classification accuracy, while the last measure was adopted to analyze the speed of computation. The processing times were recorded for videos having a resolution of 720×420 on a PC powered by Intel Core i7-2600 3.40 GHz CPU with 16 GB RAM. To ensure an unbiased computing platform, the uses of graphics processing units (GPUs) were disabled.

The evaluation is based on a non-linear ranking system with the average rank computed over individual performance metrics and across both the data sets. The higher the scores of the metrics, the better is the performance and the rank. The overall quantitative results based on all sequences of the CD.net, and the BMC data set are summarized in Table-I. The performance scores in each evaluation category were converted to ordinal ranks, which are included in the parentheses alongside. The boldface values for each of the considered metrics indicate the best results achieved by the compared methods.

Fig. 4. Qualitative results are shown row-wise with two sequences selected from each category of the CD.net dataset [32]. Starting from the top, the first row (a) illustrates results from the *pedestrians* (frame# 471) sequence of the *baseline* category. Row (b) illustrates results from the *canoe* (frame# 993) sequence of the *dynamic background* category. Row (c) illustrates results of the *boulevard* (frame# 1197) from the category *camera jitter*. Row (d) depicts results from the *street light* (frame# 2185) sequence of the *intermittent object motion* category. Row (e) contains results from the *cubicle* (frame# 4987) sequence of the *shadow* category. Finally, row (f) illustrates results from the *library* (frame# 2735) sequence of the *thermal* category. The input frames, the ground-truth masks, and the output of each algorithm are arranged column-wise as indicated at the top of the figure.

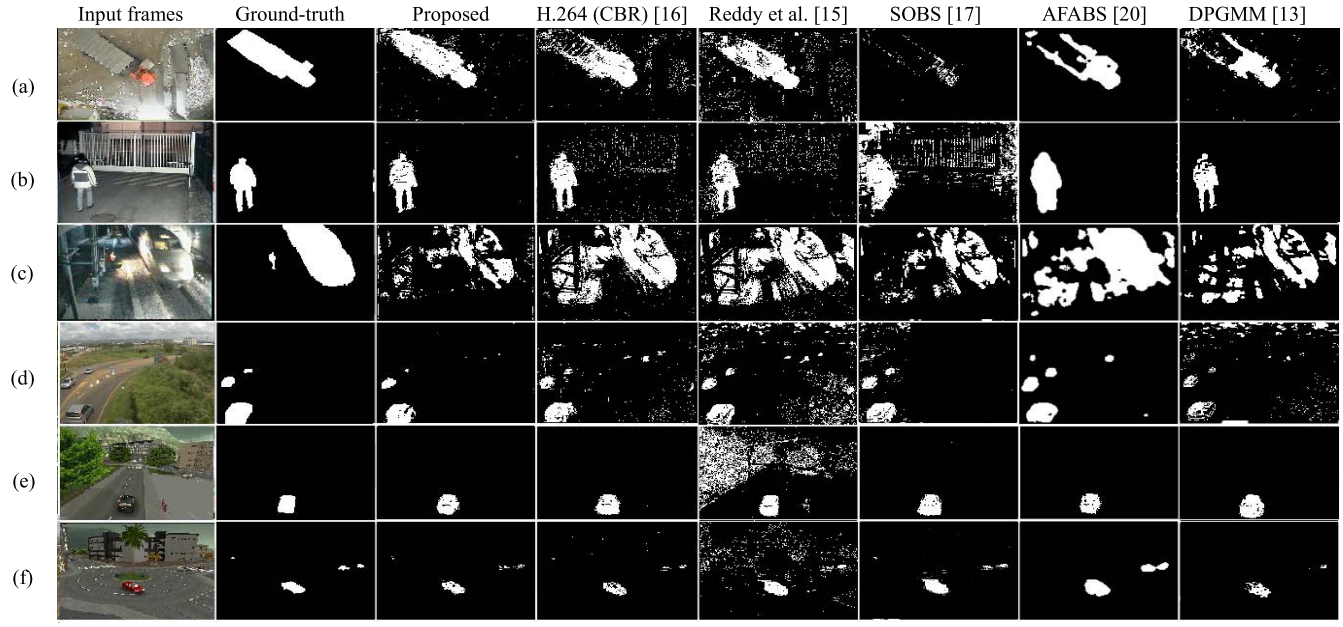

Fig. 5. Qualitative results on selected evaluation sequences of the BMC dataset [33]. Starting from the top, the first four rows, i.e., (a)-(d) correspond to real sequences *Video2* (frame# 1249), *Video4* (frame# 213), *Video7* (frame# 258), and *Video8* (frame# 130) respectively. Rows (e) and (f) illustrate results for synthetic sequences 112 (frame# 300) and 222 (frame# 645). The input frames, the ground-truth masks, and the output of each algorithm are arranged column-wise as indicated at the top of the figure.

Finally, for each method, the average rank has been obtained by computing the arithmetic mean achieved according to each single metric.

Prior to a formal evaluation, it is important to realize that compression essentially introduces visual artefacts. As a result, foreground segmentation directly using features of compressed video is accurate to the extent that can only be obtained after transcoding. Despite the above fact, it is observed that the proposed method obtained better results when it came to the overall performance comparison in terms of segmentation accuracy and processing speed. Table-II, on the other hand, analyzes the segmentation performance (using F-measure) on various sequence categories. Our performance is best demonstrated in the baseline and the dynamic background category,

TABLE I

OVERALL QUANTITATIVE COMPARISON ON CD.NET AND BMC DATA SETS (PERFORMANCE SCORES ARE AVERAGE OF ALL SEQUENCES IN THE CORRESPONDING DATA SETS)

Method	Average Recall		Average Precision		Average F-Measure		Average speed (fps)	**Average Rank**
	CD.net	BMC	CD.net	BMC	CD.net	BMC		
Proposed ST-HBF	0.7997 (3)	0.7905 (2)	**0.8339** (1)	**0.9228** (1)	**0.8164** (1)	**0.8515** (1)	**169.1** (1)	**1.4286**
DPGMM [13]	**0.8275** (1)	**0.8027** (1)	0.7928 (3)	0.8018 (3)	0.8098 (2)	0.8022 (2)	6.9 (5)	2.4286
Reddy et al. [15]	0.7935 (4)	0.7811 (3)	0.7294 (5)	0.7991 (4)	0.7601 (4)	0.7900 (3)	9.2 (3)	3.7143
SOBS [17]	0.8016 (2)	0.7316 (5)	0.7316 (4)	0.7916 (5)	0.7650 (3)	0.7604 (4)	7.3 (4)	3.8571
H.264 (CBR) [16]	0.5899 (6)	0.6612 (6)	0.7986 (2)	0.8916 (2)	0.6786 (6)	0.7593 (5)	151.1 (2)	4.1429
AFABS [20]	0.7577 (5)	0.7391 (4)	0.7191 (6)	0.7794 (6)	0.7379 (5)	0.7587 (6)	5.2 (6)	5.4286

TABLE II

QUANTITATIVE EVALUATION ON CD.NET DATA SET (USING AVERAGE F-MEASURE FOR EACH SEQUENCE CATEGORY)

Method	Baseline	Camera Jitter	Dynamic Background	Intermittent Object Motion	Shadow	Thermal	**Average Rank**
Proposed ST-HBF	**0.9342** (1)	0.7124 (3)	**0.8283** (1)	0.5351 (3)	0.8644 (2)	0.7353 (2)	**2.0000**
DPGMM [13]	0.9333 (2)	0.7477 (2)	0.8137 (2)	0.5418 (2)	0.8128 (3)	**0.8134** (1)	**2.0000**
Reddy et al. [15]	0.9206 (4)	**0.7784** (1)	0.6823 (4)	0.4955 (5)	**0.8671** (1)	0.5957 (6)	3.5000
SOBS [17]	0.9286 (3)	0.7051 (4)	0.6686 (5)	**0.5918** (1)	0.7784 (6)	0.6923 (4)	3.8333
H.264 (CBR) [16]	0.8349 (5)	0.6737 (5)	0.7829 (3)	0.4679 (6)	0.7871 (5)	0.6328 (5)	4.8333
AFABS [20]	0.7294 (6)	0.6036 (6)	0.5764 (6)	0.5334 (4)	0.8103 (4)	0.7183 (3)	4.8333

which includes scenes depicting strong background motion: boats on shimmering water (result from canoe sequence shown in Fig. 4-b), cars and pedestrians passing through background containing a fountain, waving tree, etc.

The proposed method is found to be significantly faster than the normal streaming rate, i.e., 25-30 frames/second. It is also overwhelmingly faster compared to any of the related pixel-based methods. As discussed in sections III-C, III-D, and III-G, the computation required for each CTU in terms of the number of comparisons, additions/subtractions, and multiplications/divisions cost up to a constant factor. Consequently, the complexity of the proposed method evaluates to be of the order $(c_1 \cdot \text{card}(S_t) + c_2 \cdot (C - \text{card}(S_t)))$, where $\text{card}(S_t)$ (i.e., cardinality of S_t) is the number of selected CTUs requiring pixel level processing in the given frame t, C is the total number of CTUs per frame, and c_1, c_2 are constants. With $\text{card}(S_t) \ll C$ being the dominant factor, the running time scales *linearly* with $\text{card}(S_t)$, incurring only a negligible overhead in addition to the regular decoding cost claimed by each frame. Given today's limitation on network bandwidth and overwhelmingly high computational demands in real-time, the qualitative and quantitative analyses significantly favor the proposed method in comparison to the SoA.

C. The Effect of CTU Size on Performance

It is worth mentioning that the choice of the size $L \times L$ of each CTU (i.e., whether $L = 16$, 32, or 64) is entirely an encoding decision. For wider resolution videos, HEVC benefits from using larger CTU sizes in terms of bandwidth. To find the effect of CTU size on segmentation accuracy and processing speed, two sequences form the CD.net data set, i.e., *Fall* and *PETS2006*, were encoded under three different CTU sizes $L = \{16, 32, 64\}$. Experimentally, it is observed that the average processing speed increases significantly as L grows from 16 to 64. This is perfectly in line with the earlier discussion on computational complexity as the number of CTUs having a larger size decreases in case the input video resolution remains fixed. However, we observed a minor dip in segmentation accuracy (assessed in terms of F-measure) for larger CTU sizes. This is because the segmentation map obtained with large CTUs corresponds to a very coarse approximation of the foreground-background boundary. Fig. 6 illustrates the variation of (a) average processing speed, and (b) F-measure obtained for CTU block sizes $L = 16$, 32, and 64.

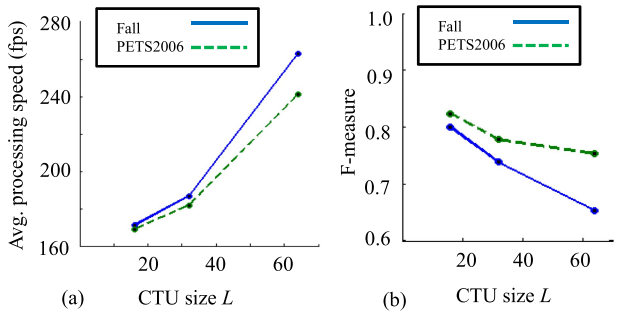

Fig. 6. Variation of (a) avg. processing speed and (b) F-measure against the CTU block size $L = 16$, 32, and 64.

V. CONCLUSION

In this paper, a method for extracting foreground objects using novel CTU features of HEVC compressed video is proposed. The method exploits the fact that compressed HEVC video is essentially a source of highly de-correlated data having two features that sufficiently describe each CTU block. The proposed method delivers a segmentation performance comparable to those of the SoA methods while maintaining low computational requirements that are affordable well within real-time constraints.

ACKNOWLEDGMENT

The first author wishes to thank Bholanath Dey and Jharna Dey for financing the overlength page charges.

Malay K. Kundu acknowledges the Indian National Academy of Engineering (INAE) for their support through INAE Distinguished Professor fellowship.

REFERENCES

[1] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: A systematic survey," *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 294–307, Mar. 2005.

[2] T. Huang, "Surveillance video: The biggest big data," *Comput. Now*, vol. 7, no. 2, Feb. 2014. [Online]. Available: http://www.computer.org/portal/web/computingnow/archive/february2014

[3] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[4] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Comput. Sci. Rev.*, vols. 11–12, pp. 31–66, May 2014.

[5] N. J. B. McFarlane and C. P. Schofield, "Segmentation and tracking of piglets in images," *Mach. Vis. Appl.*, vol. 8, no. 3, pp. 187–193, 1995.

[6] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, Jul. 1997.

[7] N. Friedman and S. Russell, "Image segmentation in video sequences: A probabilistic approach," in *Proc. 13th Conf. Uncertainty Artif. Intell.*, 1997, pp. 175–181.

[8] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 1999, pp. 246–252.

[9] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Aug. 2004, pp. 28–31.

[10] Z. Chen and T. Ellis, "Self-adaptive Gaussian mixture model for urban traffic monitoring system," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1769–1776.

[11] Y. Wang, K.-F. Loe, and J.-K. Wu, "A dynamic conditional random field model for foreground and shadow segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 2, pp. 279–289, Feb. 2006.

[12] D. Mukherjee, Q. M. J. Wu, and T. M. Nguyen, "Multiresolution based Gaussian mixture model for background suppression," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 5022–5035, Dec. 2013.

[13] T. S. F. Haines and T. Xiang, "Background subtraction with Dirichlet process mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 670–683, Apr. 2014.

[14] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proc. Eur. Conf. Comput. Vis.*, 2000, pp. 751–767.

[15] V. Reddy, C. Sanderson, and B. C. Lovell, "Improved foreground detection via block-based classifier cascade with probabilistic decision integration," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 1, pp. 83–93, Jan. 2013.

[16] B. Dey and M. K. Kundu, "Robust background subtraction for network surveillance in H.264 streaming video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1695–1703, Oct. 2013.

[17] L. Maddalena and A. Petrosino, "The SOBS algorithm: What are the limits?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 21–26.

[18] T. Bouwmans and E. H. Zahzah, "Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance," *Comput. Vis. Image Understand.*, vol. 122, pp. 22–34, May 2014.

[19] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, R. G. Baraniuk, and R. Chellappa, "Compressive sensing for background subtraction," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 155–168.

[20] P. Chiranjeevi and S. Sengupta, "Neighborhood supported model level fuzzy aggregation for moving object segmentation," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 645–657, Feb. 2014.

[21] C. Chen, J. Cai, W. Lin, and G. Shi, "Incremental low-rank and sparse decomposition for compressing videos captured by fixed cameras," *J. Vis. Commun. Image Represent.*, vol. 26, pp. 338–348, Jan. 2015.

[22] L. Zhao, X. Zhang, Y. Tian, R. Wang, and T. Huang, "A background proportion adaptive Lagrange multiplier selection method for surveillance video on HEVC," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2013, pp. 1–6.

[23] X. Guo, S. Li, and X. Cao, "Motion matters: A novel framework for compressing surveillance videos," in *Proc. ACM Int. Conf. Multimedia*, Oct. 2013, pp. 549–552.

[24] C. Chen, J. Cai, W. Lin, and G. Shi, "Surveillance video coding via low-rank and sparse decomposition," in *Proc. 20th ACM Int. Conf. Multimedia*, Oct. 2012, pp. 713–716.

[25] X. Zhang, T. Huang, Y. Tian, and W. Gao, "Background-modeling-based adaptive prediction for surveillance video coding," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 769–784, Feb. 2014.

[26] X. Zhang, Y. Tian, T. Huang, S. Dong, and W. Gao, "Optimizing the hierarchical prediction and coding in HEVC for surveillance and conference videos with background modeling," *IEEE Trans. Image Process.*, vol. 23, no. 10, pp. 4511–4526, Oct. 2014.

[27] J. Nightingale, Q. Wang, C. Grecos, and S. R. Goma, "Deriving video content type from HEVC bitstream semantics," *Proc. SPIE*, vol. 9139, p. 913902, May 2014.

[28] A. Fuldseth, G. Bjøntegaard, M. Budagavi, and V. Sze, *CE10: Core Transform Design for HEVC*, document JCTVC-G495, Nov. 2011.

[29] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1984.

[30] E. Y. Lam and J. W. Goodman, "A mathematical analysis of the DCT coefficient distributions for images," *IEEE Trans. Image Process.*, vol. 9, no. 10, pp. 1661–1666, Oct. 2000.

[31] A. N. Akansu and R. A. Haddad, "Factorization of the coefficient variance matrix in orthogonal transforms," *IEEE Trans. Signal Process.*, vol. 39, no. 3, pp. 714–718, Mar. 1991.

[32] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "A novel video dataset for change detection benchmarking," *IEEE Trans. Image Process.*, vol. 23, no. 11, pp. 4663–4679, Nov. 2014.

[33] A. Vacavant, T. Chateau, A. Wilhelm, and L. Lequièvre, "A benchmark dataset for foreground/background extraction," in *Proc. Asian Conf. Comput. Vis. Workshops, Background Models Challenge*, vol. 7728. Nov. 2012, pp. 291–300.

[34] F. Bellard. (Apr. 26, 2002). *FFmpeg*. [Online]. Available: http://ffmpeg.org

[35] D.-S. Pham, O. Arandjelovic, and S. Venkatesh, "Detection of dynamic background due to swaying movements from motion features," *IEEE Trans. Image Process*, vol. 24, no. 1, pp. 332–344, Jan. 2015.

Bhaskar Dey received the B.Tech. degree in information technology from the University of Kalyani, Kalyani, India, in 2007, and the M.Tech. degree in information technology from the University of Calcutta, Kolkata, India, in 2009.

He is currently pursuing the Ph.D. degree with the Center for Soft Computing Research, Indian Statistical Institute, Kolkata. His current research interests include statistical signal processing, video and image analysis, machine learning, and pattern recognition.

Malay K. Kundu (M'90–SM'99) received the B.Tech., M.Tech., and Ph.D. (Tech.) degrees in radio physics and electronics from the University of Calcutta, Kolkata, India.

He joined the Indian Statistical Institute (ISI), Kolkata, India, in 1982, as a Faculty Member. He is currently a Co-Principal Investigator of the Center for Soft Computing Research: A National Facility (established by Government of India) at ISI, Kolkata. He superannuated from the service of the institute as Professor (HAG) in 2013. He is the Indian National Academy of Engineering (INAE) Distinguished Professor with the Machine Intelligence Unit of this Institute. His current research interest includes digital image processing, machine learning, content based image retrieval, digital watermarking, wavelets, soft computing and computer vision. He has contributed five edited book volumes, about 150 research papers in prestigious archival journals, international refereed conferences, and in the edited monograph volumes. He holds nine U.S patents, two international, and two E.U patents.

Dr. Kundu is a fellow of the International Association for Pattern Recognition, USA, the Indian National Academy of Engineering, the National Academy of Sciences, and the Institute of Electronics and Telecommunication Engineers, India. He is the Founding Life Member and Vice President of the Indian Unit for Pattern Recognition and Artificial Intelligence (IUPRAI). He was a recipient of the Sir. J. C. Bose Memorial Award of the Institute of Electronics and Telecommunication Engineers, India, in 1986, and the prestigious VASVIK Award for industrial research in the field of electronic sciences and technology in 1999.