Enhanced Macroblock Features for Dynamic Background Modeling in H.264/AVC Video Encoded at Low-Bitrate

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

***Abstract*— While H.264 is a well-established standard for video surveillance, its High profile implementation, in particular, has the unique capabilities that pack more visual detail into a given bitrate. Several algorithms exist that detect moving objects from H.264 bitstream, most of which end up wrongly classifying the non-stationary or dynamic components (e.g. jitter camera, waving trees, ripples, etc.) of the background as foreground. Moreover, the coarse quantization schemes adopted at constrained bitrates pose new challenges for direct identification of moving objects/targets from the bitstream. This paper focuses on dynamic background modeling for H.264 video encoded at very low bitrates. To this end, enhanced features of the fundamental coding unit, i.e., the macroblock are proposed. Based on the temporal statistics of macroblock features gathered from initial frames, macroblocks potentially containing parts of moving objects are selected in subsequent frames. The selected macroblocks constitute a coarse segmentation map of the foreground at the macroblock-level. Finally, pixel-level segmentation of the foreground is obtained by comparing pixels constituting the selected macroblocks with the co-located counterparts of a background frame. Experimental results showing comparison on bitstreams encoded at strikingly low bitrates are obtained over a diverse set of standardized surveillance sequences.**

Index Terms— **background subtraction, H.264/AVC, quantization, transform coding, video surveillance.**

I. Introduction

BACKGROUND modeling [1] is a common technique used for segmenting out regions of interest (ROI) in a scene for applications such as surveillance, tracking; etc. State-of-the-art (SoA) methods can be broadly grouped into pixel-domain approaches and compressed-domain approaches. A classical pixel-domain algorithm is built around the philosophy that a background model is independently estimated at each pixel-location for a series of uncompressed image frames at the temporal axis. However, most visual information today is stored/transmitted in a compressed format (such as M-JPEG, MPEG-x, H.26x, etc.). Therefore, pixel-domain methods, including those from the current SoA [2]-[3], involve a complete decompression of the input bitstream thereby requiring a heavy computational overhead and storage space before they are applied. On the other hand, the newly developed compressed-domain methods necessitate only partial decoding of bitstream semantics to approximate block regions in each frame containing potential object motion.

Owing to its impressive compression efficiency, H.264 / MPEG-4 Advanced Video Coding (AVC) [4]-[5] is an industry standard for a vast spectrum of applications, including video surveillance [6]. Zeng et al. [7] proposed an algorithm to segment moving objects from the sparse motion vector (MV) field under the Markovian assumption. Without global motion compensation (GMC), a limitation of this approach is that it is only applicable to sequences with stationary background. Solana-Cipres et al. [8] used fuzzy linguistic concepts involving MVs and macroblock (MB) decision modes to classify MBs into foreground or background. Chen et al. [9] segmented moving objects from H.264 video using GMC and Markov Random Field classification. In [10]-[11], results are based on the premise that MBs containing complex motion consume more coding bits than those corresponding to static background. This leads to poor segmentation performance on sequences encoded at low bitrate, as the standard rate-distortion optimization (RDO) methods negotiate encoding bits for substantial distortion/mismatch from the source signal, especially in the complex regions of a scene. It may be noted that the number of coding bits allocated to a particular MB is dependent by quantization of residual signals. Therefore, in this paper, quantization step sizes of individual transform coefficients are factored into the derivation of MB features. The significant contributions to the SoA are discussed as follows:

1) It is known that video encoders can choose to improve image quality in any encoding situation where decisions are made that affect both file size and quality simultaneously. The classical method of making encoding decisions is for the encoder to choose the result which yields the highest quality output image for a given bandwidth. Under low and constrained bandwidth, the

Manuscript received February 10, 2016; revised August 31, 2016; accepted September 27, 2016.

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/TCSVT.2016.XXXXXXX

choice that the encoder makes, although require more bits in complex segments of an image still suffer from quality benefit (as determined by quantization) compared to less complex areas. As a consequence, the methods [10]-[11] which depend *solely* on the number of *encoding bits* fail to detect motion/activity when the bitrate is severely constrained and rate-distortion optimization is enabled. Therefore, in the current work, enhanced MB features are proposed which use both encoding bits *as well as* the quantization step-sizes of individual coefficients in a MB. This is particularly important as the quantization parameters vary widely between simple and complex sections of an encoded image at lower bitrates;

2) The proposed method is a two-stage *hybrid* segmentation where MBs covering probable foreground regions are first identified. A novel thresholding technique is also used to select the candidate MBs corresponding to foreground objects.
3) In the second stage, pixels constituting the selected MBs are classified into foreground or background by comparing their counterparts with those of a background model. It may be noted that the presence of shadows usually causes false classification, leading to various unwanted behavior such as object shape distortion and object merging. Therefore, instead of using the native YCbCr color space, pixel differences are estimated using the CIELUV color space because of its *perceptual uniformity* of color differences. Experimental results [12] on real-life surveillance videos have shown that the CIELUV color space is the most efficient in modeling cast shadows.

The subsequent sections of this paper are organized as follows. Section-II provides an overview of the proposed method. The feature extraction process is described in Section-III. Section-IV and Section-V summarize the experimental results and the conclusion respectively.

II. The Enhanced Macroblock Features

As mentioned earlier, the existing MB features depend only on the number of *encoding bits* and hence fail to detect motion/activity when the bitrate is severely constrained and rate-distortion optimization is enabled. Therefore, in this section enhanced MB features are proposed which takes, both - the encoding bits as well as the effective quantization step sizes of individual coefficients.

A. Preliminaries

In compressed video, MVs are used to predict temporally correlated patches of image from neighboring frames, while the residual error, i.e., the difference between the predicted and the target patch, is subjected to signal transforms to reduce the spatial redundancy. Using a compressed video analyzer[1], it was verified that the MB regions corresponding to moving objects contain MVs and transform coefficients of significantly larger magnitudes than those corresponding to the scene background. Hence, the *signal energies* associated with the quantized transform coefficients and the MVs of a given MB, heretofore designated as F_1 and F_2 respectively, are adopted as the features necessary for moving object segmentation. We assume K MBs in each frame, which are addressed numerically using a MB index $idx \in \{1,2,\ldots, K\text{-}1\}$ in the raster-scan order starting with idx=0 for the MB at the top-left-hand corner. Formally, the feature vector corresponding to the MB at location idx in frame t is denoted as the pattern (or feature) vector

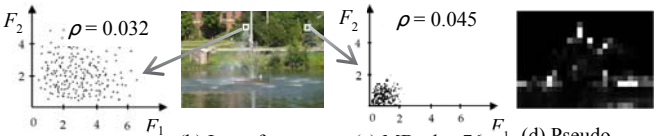
$$\vec{F}_{t,idx} = (F_1, \quad F_2)^T,$$

where $F_1, F_2 \in \mathbb{R}_{\geq 0}$, the set on non-negative reals.

It may be noted that the determinant of covariance $|\Sigma_{idx}|$ between F_1 and F_2 for all MBs at location idx, reflects the measure of randomness or 'dynamic' entity present at that location in the scene. To illustrate this fact, we computed the (temporal) mean $\vec{\mu}_{idx}$ and the covariance Σ_{idx} for all $idx \in \{1,2,\ldots, K\text{-}1\}$ from a set of initial 500 frames of containing randomness at the background. Fig. 1 shows the scatterplots of F_1-F_2 feature space for MBs at static as well as dynamic locations of *fountain02* sequence. We observe a higher $|\Sigma_{idx}|$ value (Fig. 1-a) for the MB regions corresponding to 'dynamic' location compared to that of a 'static' location (Fig. 1-c). The correlation coefficient (ρ) for both locations, however, are very close to zero, illustrating the fact that F_1 and F_2 are statistically uncorrelated. Fig. 1-b and Fig. 1-d shows the input frame and the corresponding grayscale pseudo-image obtained by scaling $\{\Sigma_{idx}\}_{idx=0}^{K-1}$ to [0, 255]. In the pseudo-image, the location of MBs with higher $|\Sigma_{idx}|$, i.e., brighter blocks, strongly correlate with the presence of dynamic content while those having a lower value of $|\Sigma_{idx}|$, i.e., darker blocks, correspond to static areas of the background.

The MB feature F_1, as introduced above, is computed as the signal energy associated with its quantized transform coefficients. Computation of F_1 using decoded transform coefficients is computationally challenging. Therefore, F_1 is statistically predicted using the number of transform coding bits, and the quantization parameters QP^C (with C=Y for the luma component, and C=Cb, Cr for the chroma components) of a MB. These values of the parameters are parsed, in real-time, from the H.264 bitstream coding syntax.

It may here be noted that the H.264/AVC standard adopted multiple transform block sizes (4×4 and 8×8) to improve the coding efficiency. The transforms were designed as approximations to the inverse discrete cosine transform (DCT) with emphasis on minimizing the number of arithmetic operations. These transforms had large variations of the norm



(a) MB *idx* =66 (dynamic) (b) Input frame #500 (c) MB *idx* =76 (static) (d) Pseudo-image

Fig. 1. Scatterplots correspond to two different MBs depicting (a) dynamic content and (c) static content. (b) an input frame selected from *fountain02* sequence with MBs demarcated in white,and (d) the corresponding pseudo-image of $|\Sigma_{idx}|$scaled to [0, 255].

[1] CodecVisa ®: www.codecian.com

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2614984, IEEE Transactions on Circuits and Systems for Video Technology

of the basis vectors. As a consequence of this, non-flat default quantization matrices were specified to compensate for different norms of the basis vectors [13]. However, uniform quantization was assumed in [11], which causes its performance to suffer significantly at constrained low bitrates. Therefore, we derive independently, the relation between quantization step size Q and quantization parameter QP^{C} effective for each coefficient in 4×4 and 8×8 transform blocks, and use it to derive the expression of F_1.

B. Relation between QP^{C} and Q in a 4×4 coefficient block

The transform process in H.264 is implemented in conjunction with quantization, to enable a division-free process involving 16-bit addition, multiplication, and binary-shift operations on integers. Considering a 2-dimensional DCT of a 4×4 block of input signal X, the integer approximation used is the matrix product

$$Y = A \cdot X \cdot A^{\mathrm{T}}, \tag{1}$$

where A denotes the core-transform matrix [13]. Since the basis vectors of A have different norms, Y is normalized using the matrix

$$E = \mathrm{normr}(A) \circ \mathrm{normr}(A)^{\mathrm{T}}, \tag{2}$$

where

1) Function normr(A) returns a matrix of the same size as input A, with the output matrix having each element equal to the reciprocal of the norm of the corresponding row vector in A; and
2) ∘ denotes the Hadamard product of two matrices

Substituting the value of A in (2), we have

$$E = \begin{bmatrix} 1/4 & 1/2\sqrt{10} & 1/4 & 1/2\sqrt{10} \\ 1/2\sqrt{10} & 1/10 & 1/2\sqrt{10} & 1/10 \\ 1/4 & 1/2\sqrt{10} & 1/4 & 1/2\sqrt{10} \\ 1/2\sqrt{10} & 1/10 & 1/2\sqrt{10} & 1/10 \end{bmatrix}.$$

Using E, the required process that yields the resultant block Z of normalized coefficients is [13]

$$Z = Y \circ E. \tag{3}$$

As mentioned earlier, the above process is implemented in conjunction with quantization as [13]

$$\begin{aligned} k_{i,j} &= \mathrm{round}(Z_{i,j}/Q_{i,j}) = \mathrm{round}(Y_{i,j}E_{i,j}/Q_{i,j}) \quad [\because Z = Y \circ E] \\ &= \mathrm{round}(Y_{i,j}S_{i,j}/2^{L}) = \mathrm{sgn}(Y_{i,j})((|Y_{i,j}|S_{i,j} + f) \gg L), \end{aligned} \tag{4}$$

where subscripts $i, j = 0,\ldots,3$ are denote the element at the (i+1)th row and the (j+1)th column of the corresponding 4×4 matrix, and $L = 15 + \lfloor QP^{C}/6 \rfloor$. S is specified by the quantization matrix M as

$$S_{i,j} = \begin{cases} M_{m,0} & \text{for } (i \bmod 2, j \bmod 2) = (0,0), \\ M_{m,1} & \text{for } (i \bmod 2, j \bmod 2) = (1,1), \\ M_{m,2} & \text{otherwise;} \end{cases}$$

where $m = QP^{\mathrm{C}} \bmod 6$, and

$$M = \begin{bmatrix} 13107 & 11916 & 10082 & 9362 & 8192 & 7282 \\ 5243 & 4660 & 4194 & 3647 & 3355 & 2893 \\ 8066 & 7490 & 6554 & 5825 & 5243 & 4559 \end{bmatrix}^{\mathrm{T}}.$$

From (4), we have

$$E_{i,j}/Q_{i,j} = S_{i,j}/2^{L},$$

which on rearrangement of terms gives

$$Q_{i,j} = (E_{i,j}/S_{i,j})2^{L}. \tag{5}$$

Finally, substituting the values of L, $E_{i,j}$, and $S_{i,j}$ in (5), the required relation between QP^{C} and Q for 4×4 coefficient block (say $Q4$) is obtained as

$$Q4^{\mathrm{C}}_{i,j} = \begin{cases} R_{m,0} & \text{for } (i \bmod 2, j \bmod 2) = (0,0), \\ R_{m,1} & \text{for } (i \bmod 2, j \bmod 2) = (1,1), \\ R_{m,2} & \text{otherwise;} \end{cases} \tag{6}$$

where

$$R = 2^{\left\lfloor \frac{QP^{\mathrm{C}}}{6} \right\rfloor} \begin{bmatrix} \frac{2^{13}}{13107} & \frac{2^{11}}{2979} & \frac{2^{12}}{5041} & \frac{2^{12}}{4681} & 1 & \frac{2^{12}}{3641} \\ \frac{2^{14}}{26215} & \frac{2^{12}}{5825} & \frac{2^{13}}{10485} & \frac{2^{14}}{18235} & \frac{2^{14}}{16775} & \frac{2^{14}}{14465} \\ \frac{2^{13}}{4033\sqrt{10}} & \frac{2^{13}}{3745\sqrt{10}} & \frac{2^{13}}{3277\sqrt{10}} & \frac{2^{14}}{5825\sqrt{10}} & \frac{2^{14}}{5243\sqrt{10}} & \frac{2^{14}}{4559\sqrt{10}} \end{bmatrix}^{\mathrm{T}},$$

and superscript C = Y, Cb, Cr depending on the color component associated with the input block X.

C. Relation between QP^{C} and Q in a 8×8 coefficient block

The adaptive 8×8 transform is optionally supported in High profiles. When selected for coding a MB, the 8×8 transform is applied to the luma signal only, while the chroma components are subjected to 4×4 transforms as usual. The core integer transform of an 8×8 block of input signal X is performed according to (2), where

$$A = \begin{bmatrix} 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 12 & 10 & 6 & 3 & -3 & -6 & -10 & -12 \\ 8 & 4 & -4 & -8 & -8 & -4 & 4 & 8 \\ 10 & -3 & -12 & -6 & 6 & 12 & 3 & -10 \\ 8 & -8 & -8 & 8 & 8 & -8 & -8 & 8 \\ 6 & -12 & 3 & 10 & 10 & -3 & 12 & -6 \\ 4 & -8 & 8 & -4 & -4 & 8 & -8 & 4 \\ 3 & -6 & 10 & -12 & 12 & -10 & 6 & -3 \end{bmatrix} \cdot 1/8$$

as specified by [14]. Substituting the value of A in (2), the matrix E for 8×8 block transforms is obtained as

$$E = \begin{bmatrix} 1/8 & 2/17 & 1/2\sqrt{10} & 2/17 & \cdots \\ 2/17 & 32/289 & 8/17\sqrt{10} & 32/289 & \\ 1/2\sqrt{10} & 8/17\sqrt{10} & 1/5 & 8/17\sqrt{10} & \\ 2/17 & 32/289 & 8/17\sqrt{10} & 32/289 & \\ \vdots & & & \ddots & \end{bmatrix}_{8\times 8}.$$

Only the top-left quadrant is shown, while the remaining three quadrants are identical. The quantization of an 8×8 block of transform coefficients is implemented according to (5) with the exception that $L = 16 + \lfloor QP^{C}/6 \rfloor$,

$$S_{i,j} = \begin{cases} M_{m,0} & \text{for } (i \bmod 4, j \bmod 4) = (0,0), \\ M_{m,1} & \text{for } (i \bmod 2, j \bmod 2) = (1,1), \\ M_{m,2} & \text{for } (i \bmod 4, j \bmod 4) = (2,2), \\ M_{m,3} & \text{for } (i \bmod 4, j \bmod 2) = (0,1) \\ & \text{or } (i \bmod 2, j \bmod 4) = (1,0), \\ M_{m,4} & \text{for } (i \bmod 4, j \bmod 4) \in \{(0,2),(2,0)\}, \\ M_{m,5} & \text{otherwise;} \end{cases}$$

where subscripts $i, j = 0,\ldots,7$ denote the element at (i+1)th row and (j+1)th column of the corresponding 8×8 matrix, and

$$M = \begin{bmatrix} 13107 & 11428 & 20972 & 12222 & 16777 & 15481 \\ 11916 & 10826 & 19174 & 11058 & 14980 & 14290 \\ 10082 & 8943 & 15978 & 9675 & 12710 & 11985 \\ 9362 & 8228 & 14913 & 8931 & 11984 & 11259 \\ 8192 & 7346 & 13159 & 7740 & 10486 & 9777 \\ 7282 & 6428 & 11570 & 6830 & 9118 & 8640 \end{bmatrix}.$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2614984, IEEE Transactions on Circuits and Systems for Video Technology

Finally, substituting the values of L, $E_{i,j}$ and $S_{i,j}$ in (5), the required relation between QP and Q for an 8×8 coefficient block (say $Q8$) is obtained as

$$Q8^{C}_{i,j} = \begin{cases} R_{m,0} & \text{for } (i \bmod 4, j \bmod 4) = (0,0), \\ R_{m,1} & \text{for } (i \bmod 2, j \bmod 2) = (1,1), \\ R_{m,2} & \text{for } (i \bmod 4, j \bmod 4) = (2,2), \\ R_{m,3} & \text{for } (i \bmod 4, j \bmod 2) = (0,1) \\ & \text{or } (i \bmod 2, j \bmod 4) = (1,0), \\ R_{m,4} & \text{for } (i \bmod 4, j \bmod 4) \in \{(0,2),(2,0)\} \\ R_{m,5} & \text{otherwise;} \end{cases} \tag{7}$$

where

$$R = 2^{\left\lfloor \frac{QP^{C}}{6} \right\rfloor} \begin{bmatrix} \frac{2^{13}}{13107} & \frac{2^{19}}{825673} & \frac{2^{14}}{26215} & \frac{2^{16}}{103887} & \frac{2^{15}}{16777\sqrt{10}} & \frac{2^{19}}{263177\sqrt{10}} \\ \frac{2^{11}}{2979} & \frac{2^{20}}{1564357} & \frac{2^{15}}{47935} & \frac{2^{16}}{93993} & \frac{2^{13}}{3745\sqrt{10}} & \frac{2^{18}}{121465\sqrt{10}} \\ \frac{2^{12}}{5041} & \frac{2^{21}}{2584527} & \frac{2^{15}}{39945} & \frac{2^{17}}{164475} & \frac{2^{14}}{6355\sqrt{10}} & \frac{2^{19}}{203745\sqrt{10}} \\ \frac{2^{12}}{4681} & \frac{2^{19}}{594473} & \frac{2^{16}}{74565} & \frac{2^{17}}{151827} & \frac{2^{11}}{749\sqrt{10}} & \frac{2^{19}}{191403\sqrt{10}} \\ 1 & \frac{2^{20}}{1061497} & \frac{2^{16}}{65795} & \frac{2^{15}}{32895} & \frac{2^{14}}{5243\sqrt{10}} & \frac{2^{19}}{166209\sqrt{10}} \\ \frac{2^{12}}{3641} & \frac{2^{19}}{464423} & \frac{2^{15}}{28925} & \frac{2^{16}}{58055} & \frac{2^{14}}{4559\sqrt{10}} & \frac{2^{13}}{2295\sqrt{10}} \end{bmatrix},$$

and superscript C = Y for the luma component only.

D. Computation of F_1 for a given MB

The spatial-to-frequency transforms applied in H.264 is an integer approximation of the 2-dimentional DCT using only integer operations. The DCT coefficients are best modeled in literature [15] as a zero mean distribution with a Laplacian probability density function (pdf), i.e.,

$$p(z) = \frac{1}{2b}\exp\left(-|z|/b\right), z \in \mathbb{R} \tag{8}$$

where z is the value of a given DCT coefficient, and b>0 is a parameter that determines the coefficient variance. Quantization of an input coefficient z is performed as

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

where $k \in \mathbb{Z}$ represents the coefficient level that is encoded by the source encoder, Q>0 is the uniform quantization step-size, and f is a rounding offset with a value equal to $Q/3$ or $Q/6$ accordingly as the MB is intra- or inter-predicted [13]. It may be verified that all values of z in the open interval $(-Q+f, Q-f)$ are mapped to k=0. Similarly the quantization levels are mapped to k = 1, 2, 3,… for $z \in [(kQ-f), (kQ+Q-f))$ and k = −1, −2, −3,... for $z \in ((kQ-Q+f), (kQ+f)]$. Therefore, the probability $P_f(z_k)$ that a random input coefficient z is mapped to level k via the quantization process (9) is analytically expressed as

$$P_f(z_k) = \begin{cases} \int_{(kQ-Q+f)}^{(kQ+f)} p(z)dz = \dfrac{\mathrm{e}^{kQ/b}}{2}(\mathrm{e}^{Q/b}-1)\mathrm{e}^{-(Q-f)/b}; & \text{if } k < 0 \\ \int_{(Q-f)}^{(-Q+f)} p(z)dz = 1-\mathrm{e}^{-(Q-f)/b}; & \text{if } k = 0 \\ \int_{(kQ-f)}^{(kQ+Q-f)} p(z)dz = \dfrac{\mathrm{e}^{-kQ/b}}{2}(\mathrm{e}^{Q/b}-1)\mathrm{e}^{-(Q-f)/b}; & \text{if } k > 0 \end{cases} \tag{10}$$

where e = exp(1).

The inverse operation involving the reconstruction of coefficient value z_k corresponding to level k is given by

$$z_k = kQ. \tag{11}$$

Given b and Q for a particular coefficient position in a transform block (TB), the signal energy $F_{1,\text{coeff}}$ associated with it is given as

$$F_{1,\text{coeff}}(Q,b) = \sum_{k=-\infty}^{\infty} z_k^2 P_f(z_k) = \frac{Q^2\,\mathrm{e}^{f/b}(1+\mathrm{e}^{Q/b})}{(\mathrm{e}^{Q/b}-1)^2}. \tag{12}$$

It is theoretically based on the fact that energy of a discrete-time signal $x(n)$, which is assumed to be composed of components z_k with probability $P_f(z_k)$, is defined as $\sum_{n=-\infty}^{\infty}|x(n)|^2$. The values of Q for 4×4 (=$Q4$) and 8×8 (=$Q8$) TBs are obtained from (6) and (7), while the parameter b is evaluated by equating the number of bits (say *bits*) consumed in coding the transform coefficients of a MB with its analytical expression obtained using entropy. Recall that entropy, heretofore denoted as H_f, is a lower bound on the average number of bits required to encode a given coefficient. Using (10), H_f (in bits/coefficient) may be expressed as

$$\begin{aligned} H_f(Q,b) &= -\sum_{k=-\infty}^{+\infty} P_f(z_k)\log_2 P_f(z_k) \\ &= -(1-\mathrm{e}^{-(Q-f)/b})\log_2(1-\mathrm{e}^{-(Q-f)/b}) \\ &\quad -\frac{\mathrm{e}^{-(Q-f)/b}}{\ln 2}\left(\ln\left(\frac{\mathrm{e}^{Q/b}-1}{2}\right)-\frac{Q-f}{b}-\frac{Q\,\mathrm{e}^{Q/b}}{b(\mathrm{e}^{Q/b}-1)}\right), \end{aligned} \tag{13}$$

which is a function of Q and b.

Based on the TB sizes adopted in H.264, we formulate F_1 under two cases as follows

1) *When 4×4 transforms are used for all color components*

Considering a MB to be encoded using the popular YCbCr 4:2:0 color format, there exist sixteen 4×4 TBs of the luma component and four 4×4 blocks of each of the chroma components. The value of Q for a particular coefficient vary depending on its position in the transform block, and hence the value of H_f. Therefore, we express *bits* as the sum of all H_fs corresponding to each coefficient of a MB, i.e.

$$bits = 16\sum_{i=0}^{3}\sum_{j=0}^{3} H_f(Q4^{Y}_{i,j}, b) + 4\sum_{C=Cb,Cr}\sum_{i=0}^{3}\sum_{j=0}^{3} H_f(Q4^{C}_{i,j}, b). \tag{14}$$

In (14), the coefficients of a given MB are modeled with a single source distribution; hence, b – the distribution parameter, is constant for all TBs. The values of *bits*, B, QP^{Y}, QP^{Cb}, and QP^{Cr}, which are parsed from the MB-layer syntax in the input bitstream are used to compute $Q4^{Y}_{i,j}$, $Q4^{Cb}_{i,j}$ and $Q4^{Cr}_{i,j}$ by substituting C with Y, Cb, and Cr respectively in (6). The remaining parameter b is evaluated using standard numerical methods [22] for finding the root of non-linear equation in one unknown.

The energy associated with each 4×4 TB of a given MB is computed as the average of the value given by (12), i.e.,

$$\frac{1}{16}\sum_{i=0}^{3}\sum_{j=0}^{3} F_{1,\text{coeff}}(Q4^{\text{C}}_{i,j}, b);\ \text{C} = \text{Y, Cb, Cr.}$$

Finally, F_1 due to the entire MB is computed as the weighted average of the values obtained for individual color components. Thus, we have

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2614984, IEEE Transactions on Circuits and Systems for Video Technology

$$F_1 = \frac{1}{24}\sum_{i=0}^{3}\sum_{j=0}^{3} F_{1,\text{coeff}}(Q4_{i,j}^{\text{Y}}, b) + \frac{1}{96}\sum_{C=Cb,Cr}\sum_{i=0}^{3}\sum_{j=0}^{3} F_{1,\text{coeff}}(Q4_{i,j}^{\text{C}}, b). \quad (15)$$

The normalization factors/weights are based on the fact that in 4:2:0 sampling, the luma coefficients comprise a ratio of exactly 2/3 of the total number of coefficients of a MB, while that for the chroma components being equal to 1/6 each. It may be noted that F_1 is 'quantization-normalized' as its computation takes into account the widely varying quantization step sizes applied to individual coefficients at constrained bitrates.

2) *When 8×8 transforms are used for the luma component*

It may be noted that, encoded bitstreams using 8×8 transforms for the luma component (High profiles only) use 4×4 transforms for the chroma components. Thus, four transform blocks of each color component (but varying sizes) comprise a given MB. Therefore, we express *bits* as

$$bits = 4\left[\sum_{i=0}^{7}\sum_{j=0}^{7} H_f(Q8_{i,j}^{\text{Y}}, b) + \sum_{C=Cb,Cr}\sum_{i=0}^{3}\sum_{j=0}^{3} H_f(Q4_{i,j}^{C}, b)\right]. \quad (16)$$

The values of $Q8_{i,j}^{Y}$ are computed using (7), while those of $Q4_{i,j}^{Cb}$ and $Q4_{i,j}^{Cr}$ are obtained by replacing C with Cb and Cr respectively in (6). The unknown constant b is evaluated using numerical techniques [22] for solving non-linear equations in one unknown.

Using (12), the energy associated with an 8×8 luma TB is given as

$$\frac{1}{64}\sum_{i=0}^{7}\sum_{j=0}^{7} F_{1,\text{coeff}}(Q8_{i,j}^{\text{Y}}, b),$$

while those contributed by the chroma blocks are obtained as described for 4×4 blocks (case-1). Finally F_1 for a given MB is expressed as the normalized sum

$$F_1 = \frac{1}{96}\left[\sum_{i=0}^{7}\sum_{j=0}^{7} F_{1,\text{coeff}}(Q8_{i,j}^{\text{Y}}, b) + \sum_{C=Cb,Cr}\sum_{i=0}^{3}\sum_{j=0}^{3} F_{1,\text{coeff}}(Q4_{i,j}^{\text{C}}, b)\right]. \quad (17)$$

E. Implementation Considerations for Computation of F_1

The statistical prediction of F_1 using (15) and (17) for a given MB depends only on *bits*, QP^{Y}, QP^{Cb}, and QP^{Cr}, the values of which are non-negative integers in the range specified by the H.264 standard. The maximum number of bits allowed in the MB-layer (for bitstreams using 8-bit color sampling) is specified as 3200. Furthermore, $QP^{\text{Y}} \in \{0, 1,...,51\}$, while QP^{Cb} and QP^{Cr} are determined from Table-I based on the index denoted as qP_I. The value of qP_I is obtained as [5]

$$\text{qP}_\text{I} = \min(\max(0, QP^{\text{Y}} + \text{qP}_{\text{Offset}}), 51),$$

where $\text{qP}_{\text{Offset}} \in \{-12, -11,...,11,12\}$ is a value specified by syntax elements *chroma_qp_index_offset* (for Cb) and *second_chroma_qp_index_offset* (for Cr) in the Picture Parameter Set (PPS). As a consequence, the number of *distinct* input combinations of (*bits*, QP^{Y}, QP^{Cb}, QP^{Cr}) possible is no more than 3200×17807. In order to reduce computations at run-time, pre-computed values of F_1 for all input combinations are indexed in lookup tables. Thus, the computation required for statistical prediction of F_1 is completely replaced with simple lookup operations.

TABLE I
QP^{C} AS A FUNCTION OF qP_I

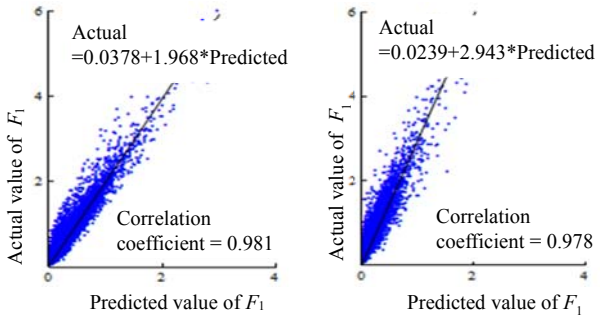
qP_I	QP^{C}	qP_I	QP^{C}	qP_I	QP^{C}	qP_I	QP^{C}
< 30	=qP_I	35	33	41	36	47	38
30	29	36	34	42	37	48	39
31	30	37	34	43	37	49	39
32	31	38	35	44	37	50	39
33	32	39	35	45	38	51	39
34	32	40	36	46	38		

Please note that the proposed formulation of F_1 is based on transform coefficient statistics using only partially decoded parameters of a MB such as *bits*, QP^{Y}, QP^{Cb}, and QP^{Cr} rather than actual coefficients decoded from compressed video. Considering the fact that signal processing transforms output exactly the same number of coefficients as the input pixels, the statistical formulation helps us to evaluate F_1 with significantly less computation as compared to that of pixel-based methods. In Section IV-A, we justify with a comparison, the significance of predicted values of F_1 against the corresponding actual values computed directly from the decoded coefficients.

F. Computation of F_2 for a given MB

F_2 is the signal energy associated with MVs of a MB. It quantifies the localized motion content of a MB partition in the current frame with respect to previously coded frames. An inter-predicted MB is composed of one of more partitions predicted from a set of previously coded reference frames. The prediction information is indicated by a set of MVs and corresponding reference frame indices. Each MV associated with an MB partition is predicted either uni-directionally (from one) or bi-directionally (from two) reference frame(s). F_2 is computed as the normalized/weighted sum of squares of the MV magnitudes of a MB. Accordingly, the weight or normalization factor associated with each MV is determined on the basis of 1) the ratio of the partition size to the total MB area it represents; and 2) the reference frame index, considering the temporal distance between the current and the referenced frame.

In H.264, a MB may be split into one, two, or four partitions using either one 16×16 partition (covering the whole



(a) *idx* = 76 (static location) (b) *idx* = 66 (dynamic location)

Fig. 2. Comparison of the predicted and the actual values of the proposed feature F_1. Regression lines are obtained by the least square error method.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2614984, IEEE Transactions on Circuits and Systems for Video Technology

MB); two 8×16 partitions; two 16×8 partitions; or four 8×8 partitions. In case of 8×8 blocks, each block is split into one, two or four *sub-partitions* (either one 8×8, two 4×8, two 8×4 or four 4×4 sub-partitions). Let

$$W_i = \text{Size of the } i\text{th partition} \big/ 256$$

denote the weight due to the *i*th partition of a MB. Furthermore, corresponding to the *i*th partition, we define

$$d_i = \begin{cases} 1; & \text{for uni-directional prediction} \\ 2; & \text{otherwise} \end{cases}$$

Corresponding to each MV (x_{ij}, y_{ij}) denoting the prediction of the *i*th partition in the *j*th direction (i.e., either forward or backward), let us denote the reference frame index as s_{ij} (where reference index 0 denotes frame one in the past/future, reference index 1 denotes frame two in the past/future, and so on). It is easy to check that the weights of all MVs of a MB sum up to *unity*. Assuming a total of *p* (maximum value being 16) partitions in a given *inter-predicted* MB, F_2 is computed as

$$F_2 = \sum_{i=1}^{p} \frac{W_i}{d_i} \sum_{j=1}^{d_i} \left[(x_{ij}^2 + y_{ij}^2) \big/ (s_{ij} + 1) \right]. \tag{18}$$

It may be noted that:

1) The value of F_2 for *intra-predicted* MBs is taken as zero as they do not encode any motion compensated information;

2) A MB may be split into a maximum of 16 partitions with each having *two* MVs in both directions. Therefore, computation of F_2 for any given MB requires no greater than 16×2×5 multiplications/divisions and 16×2×2 additions.

III. The Proposed Method

This section is divided into subsections involving background model initialization, segmentation, and update.

A. Background model initialization

Based on the proposed MB features, a 2D Gaussian pdf is fitted using the most-recent n frames. In order to avoid fitting the model from scratch for every incoming frame, a running (or on-line cumulative) estimate is computed. The background model parameters corresponding to each MB having location *idx* in frame *t*, i.e., the background mean $\vec{\mu}_{t,idx}$ and the covariance $\Sigma_{t,idx}$ are recursively computed as

$$\vec{\mu}_{t,idx} = \begin{cases} \vec{F}_{t,idx} & \text{for } t = 1 \\ \eta \vec{F}_{t,idx} + (1-\eta)\vec{\mu}_{t-1,idx} & \text{for } t \geq 2 \end{cases} \tag{19}$$

and,

$$\Sigma_{t,idx} = \begin{cases} \text{Diag}(0,0) & \text{for } t = 1 \\ \eta \left(\vec{F}_{t,idx} - \vec{\mu}_{t,idx} \right) \left(\vec{F}_{t,idx} - \vec{\mu}_{t,idx} \right)^T & \\ \quad + (1-\eta)\Sigma_{t-1,idx} & \text{for } t \geq 2 \end{cases} \tag{20}$$

where, η=0.01 is an empirically chosen parameter that determines the tradeoff between stability and quickly updates.

The background frame is initialized using the first *N*=250 frames from the input sequence. Although the set of initial frames that are used for background-model learning is ideally supposed to contain only background information, in practice, however, it is difficult to get real-world sequence not occupied by foreground objects. Hence, not a single frame is appropriate to be used as the background frame. To address this problem, a concept of the *most common frame of a scene* (McFIS) [24] has been developed for video coding using dynamic background modeling. In this paper, a recently-modified version of the McFIS generation algorithm [25] for the decoded frames has been used as the initial background frame. The parameters were initialized as follows: maximum number of models for a pixel *K*=3, learning rate α=0.1, weight ω=0.001, and variance σ=30 as used in [25].

B. Segmentation

Beginning with frame *t*=*N*+1, the segmentation process operates via a two-stage coarse-to-fine process as follows.

1) MB selection: At the coarse scale, block-level segmentation of each frame is performed by selecting a set of MBs potentially containing foreground regions. The selected MBs are those that correspond to $D_t > \alpha$, where $D_t = \sqrt{\left(\vec{F}_{t,idx} - \vec{\mu}_{t,idx}\right)^T \Sigma_{t,idx}^{-1} \left(\vec{F}_{t,idx} - \vec{\mu}_{t,idx}\right)}$ is the Mahalanobis distance of a given MB measured in units of $|\Sigma_{t,idx}|$ from $\vec{\mu}_{t,idx}$,and $\alpha = 2.8$ being a predetermined threshold.

2) Pixel-level refinement: Since real object boundaries rarely follow block boundaries, the MB-level segmentation is further refined by eliminating pixels from selected MBs that are similar to the co-located pixels in the background model. Similarity between a pair of color coordinates $X_C \equiv (L_C, U_C, V_C)$ and $X_B \equiv (L_B, U_B, V_B)$ in the CIELUV color space is ascertained if the distance

$$\Delta(X_C, X_B) = \left| \frac{L_C - L_B}{\sigma_L} \right| + \left| \frac{U_C - U_B}{\sigma_U} \right| + \left| \frac{V_C - V_B}{\sigma_V} \right| < \tau,$$

where σ_L, σ_U, and σ_V respectively denote the standard deviations of *L*, *U*, and *V* components of the corresponding background pixel. Furthermore, $\tau = 3.4$ denotes a decision threshold chosen empirically for defining the fluctuations in

TABLE II
OVERALL QUANTITATIVE COMPARISON ON H.264 ENCODED SEQUENCES OF CHANGEDETECTION.NET [16] DATA SET

Method	Average Recall		Average Precision		Average F-Measure		Average speed (fps)		Average Rank
	1000 KB/s	200 KB/s	1000 KB/s	200 KB/s	1000 KB/s	200 KB/s	1000 KB/s	200 KB/s	
Proposed (Luv)	**0.8202(1)**	**0.7470(2)**	**0.8411(1)**	**0.7932(1)**	**0.8271(1)**	**0.7579(1)**	417.8(3)	435.2(3)	**1.6250**
Proposed (YCbCr)	0.8116(2)	0.7189(3)	0.8097(2)	0.7276(3)	0.8058(2)	0.7161(3)	**436.1(1)**	**453.6(1)**	2.1250
Madalena [23]	0.8016(3)	0.8016(1)	0.7316(4)	0.7316(2)	0.7283(3)	0.7283(2)	7.3(6)	7.3(6)	3.3750
Dey et. al [11]	0.7117(5)	0.6433(5)	0.7998(3)	0.7135(4)	0.6856(4)	0.6758(5)	431.8(2)	444.3(2)	3.7500
Chen et. al [9]	0.7691(4)	0.7076(4)	0.7305(5)	0.6689(5)	0.6513(5)	0.6782(4)	115.4(5)	133.4(5)	4.8750
Poppe et. al [10]	0.7016(6)	0.6379(6)	0.6651(6)	0.5497(6)	0.6322(6)	0.5938(6)	138.7(4)	151.4(4)	5.5000

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2614984, IEEE Transactions on Circuits and Systems for Video Technology

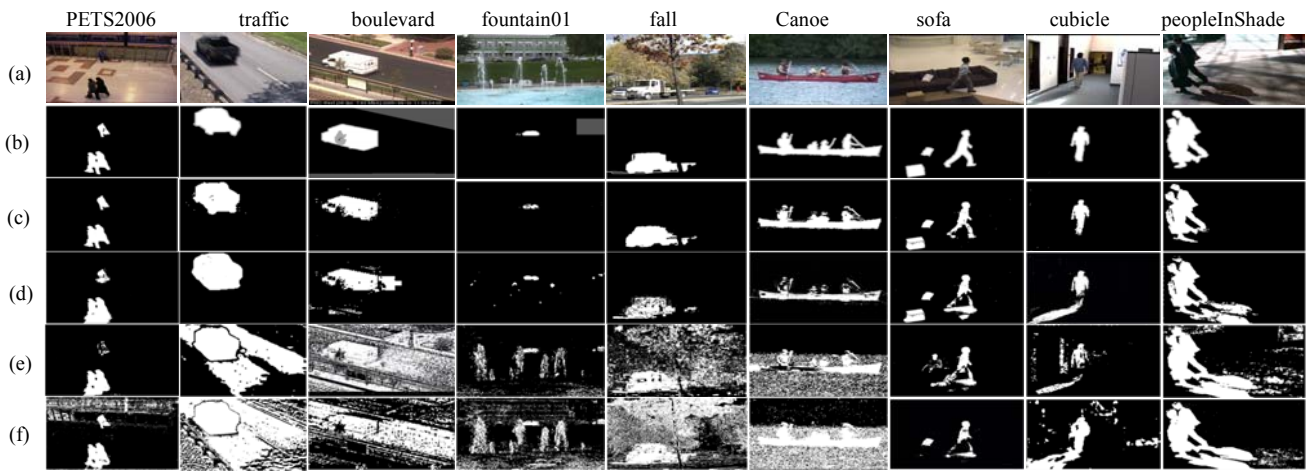


Fig. 3. Qualitative results are shown column-wise with at least one sequence selected from each category of the dataset [16]. Starting from the top, the first row (a) illustrates selected image frames. Row (b) illustrates the corresponding ground-truth masks. Row (c) and Row (d) illustrate results obtained with the proposed method using the CIELUV space and the native YCbCr space respectively. Row (e) and Row (f) demonstrate the corresponding masks obtained with [9] and [11] respectively.

luminosity and chromaticity. The value of Δ is a low-complexity approximation of the standardized Euclidean distance between X_C and X_B. The rationale behind the choice of CIELUV space is that, unlike RGB or YCbCr, it exhibits *perceptual uniformity*, i.e., for the same distance Δ computed between any two pairs of color coordinates, equal color differences are perceived by the human visual system. Experiments on real-life surveillance videos with varying illumination conditions have shown the CIELUV color space to be the most efficient. As brightness information (L component) is separated from chrominance (U and V components), the space is affected less by shadows.

C. Background model update and the choice of parameter α

Let C_t and B_t denote the input frame at t and the corresponding background frame respectively. We adopt a *selective update* policy where only pixels enclosed by the MBs that are *not* selected in the MB selection stage are used to update the current background B_t to B_{t+1} as

$$B_{t+1}(x,y) = \eta C_t(x,y) + (1-\eta) B_t(x,y), \text{ for } t > N \quad (21)$$

where $\eta = 0.01$ is the rate of update as defined earlier.

It may be noted that the proposed model essentially translates to a bivariate normal distribution $\mathcal{N}(\vec{\mu}_{t,idx}, \Sigma_{t,idx}) = (2\pi)^{-1}|\Sigma_{t,idx}|^{-1/2} \exp(-D_t^2/2)$ representing the background at the MB-level in F_1–F_2 feature space. Taking $\Sigma_{t,idx} = \text{Diag}(\sigma_1^2, \sigma_2^2)$ and $\vec{\mu}_{t,idx} = (\mu_1, \mu_2)^T$, the value of decision threshold α which correspond to 99%-prediction interval (for our background model predictions) is computed as

$$\int_{\mu_2-\alpha\sigma_2}^{\mu_2+\alpha\sigma_2} \int_{\mu_1-\alpha\sigma_1}^{\mu_1+\alpha\sigma_1} \mathcal{N}\left(\vec{\mu}_{t,idx}, \Sigma_{t,idx}\right) dF_1 dF_2 = \text{erf}^2\left(\frac{\alpha}{\sqrt{2}}\right) = 0.99. \quad (22)$$

Solving (22) for α, we get $\alpha = 2.8$.

IV. Experimental Results

In this section, the relation between statistically predicted and actually computed values of F_1 are analyzed. Subsequently, quantitative and qualitative comparisons are demonstrated on standard sequences to highlight the efficacy of the proposed method at low bitrate.

A. Comparison between the predicted and actual values of F_1

The predicted values of F_1 are validated experimentally against their actual counterparts computed directly from the decoded coefficients. Fig. 2 illustrates the comparison of F_1 for a static (*idx*=76) as well as a dynamic location (*idx*=66) of *fountain02* sequence. It is observed that the predicted values are slightly lower in magnitude compared to those which are computed directly from the decoded coefficients. This is a consequence of the fact that prediction of F_1 depends on *bits*, whose *lower bound* on the average bitrate was modeled using the entropy measure. Furthermore, the predicted value and its actual counterpart are found to be linearly correlated. As a matter of fact, the correlation coefficient was found to be greater than 0.96 for all individual sequences. Please note that the discrepancy between the predicted and the actual values, however, do not affect the MB selection process described earlier, because the Mahalanobis distance D is *invariant* under arbitrary linear transformations of the feature space. The average computation time for the predicted and the actual F_1 were also noted as 0.638 nsec and 396.7 nsec. Moreover, the average processing speeds in frames per second (fps) using the actual F_1 were found to be 14.1 fps and 12.8 fps for videos encoded at 1000 KB/s and 200 KB/s respectively. For comparison, the average processing speeds using the predicted values of F_1 are mentioned in Table-II. The overall memory requirement of the proposed method involving the actual F_1 and the predicted F_1 was analyzed to be of the order O(N) and O($N + \varepsilon$) respectively, where N is the number of MBs per frame and constant ε = 434.74MB, being the memory overhead due to look up tables. It may be also noted that O($N+\varepsilon$) is O(N); since ε is constant, i.e., the space complexity of the methods using the predicted and the actual values of F_1 are arguably the same. Fig. 4 illustrates a comparison the segmentation results obtained with the predicted and actual values of F_1. Negligible difference is observed between the segmentation results obtained with the predicted and the actual

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2614984, IEEE Transactions on Circuits and Systems for Video Technology

value of F_1. By using the predicted value of F_1, as evidenced from the results, we are able to reduce the computation time significantly without affecting the overall segmentation process.

B. Evaluation of Segmentation Results

The proposed method has been evaluated against the benchmark Changedetection.net (CD.net) dataset [16]. This dataset, unlike any other publicly available, addresses the major key challenges, especially dynamic background, accompanying a real-world surveillance scenario and includes accurate ground truth masks for each frame. The proposed method was implemented in C and patched into the original H.264 decoding module of FFmpeg [17]. The source was built on 64-bit Windows platform.

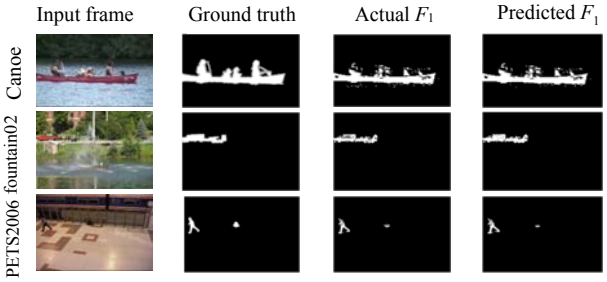


Fig. 4. Comparison of the final segmentation results of three sequences using the predicted and the actual values of the proposed feature F_1. No significant difference was registered between the segmentation results obtained with the predicted and the actual value of F_1.

Fig. 3 illustrates the qualitative (visual) comparison of segmentation results of nine selected sequences, viz. *PETS2006*, *traffic*, *boulevard*, *fountain01*, *fall*, *canoe*, *sofa*, *cubicle*, and *peopleInShade* encoded at 200KB/s. The sequences depict irregular camera jitter, dynamic background and moving cast shadows in the background, which are typical of a real-world surveillance scenario. Due to space constraints, segmentation results from two SoA methods [9] and [11] are shown in Fig. 3. Comparison of the results with the provided ground-truth masks (on row 2) shows that the proposed method is best able to model real-world surveillance scenarios, e.g., static background (*PETS2006*), mild camera jitter (*traffic* and *boulevard*), dynamic background (*fountain01*, *fall*, and *canoe*), as well as moving cast shadow (*sofa*, *cubicle*, and *peopleInShade*).

In order to compare the achieved results quantitatively, we consider the performance metrics, as reported in Table-II. The metrics adopted for quantitative evaluation includes, in addition to average processing speed, recall (Re)=#TP/(#TP+#FN), precision (Pr)=#TP/(#TP+#FP), and f-measure = 2×Pr×Re/(Pr+Re), where #TP, #FP, and #FN are the total number of true positives, false positives, and false negatives respectively. The processing speeds are based on the videos having a resolution of 720×420 pixels on a personal computer powered by Intel Core i7-2600 3.40 GHz CPU and 32GB RAM. The parenthesized figures appearing on the left of each performance score in Table-II indicate the rank of an algorithm in the corresponding evaluation category.

In order to demonstrate the proposed method for a wide range of bitrates, all sequences of the dataset [16] were encoded to H.264 High profile at target bitrates of 200KB/s (low bitrate) and 1000KB/s (high bitrate). It may be noted that *no fixed* quantization parameter was adopted for encoding the MBs, i.e., for each encoded MB in a frame the encoder was free to choose a different quantization parameter depending on the complexity of the scene and the target bitrate. The encoder configuration was set as follows: H.264 High profile YCbCr 4:2:0 progressive format with 8-bit color sampling, rate-distortion optimization (RDO) was enabled, and the MV range was set to [-16…16]. The streaming rate was fixed at 25 frames per second (fps). Segmentation results were obtained using CIELUV as well as the native YCbCr color space. The evaluation in Table-II is based on the average rank computed over the individual performance metrics discussed above. It is observed that the proposed method (both CIELUV and YCbCr) obtained better results as compared to the SoA compressed domain methods [9], [10], and [11]. The proposed method has also been compared with [23], which is a recent pixel-based method. Fig. 5 further demonstrates the impact of low bitrate encoding on the segmentation performance (accessed in terms of F-measure) on two sequences selected from the dataset. Sequences fall and canoe (depicting dynamic background) were encoded with target bitrates of 200 KB/s, 400 KB/s, 600 KB/s, 800 KB/s, and 1000 KB/s. It is observed that the proposed method outperformed the SoA methods in terms of segmentation accuracy, especially at lower bitrates. Evidently, with the enhanced MB features, the proposed method is better able to model the background dynamics under variable quantization schemes adopted under a wide range of bitrates.

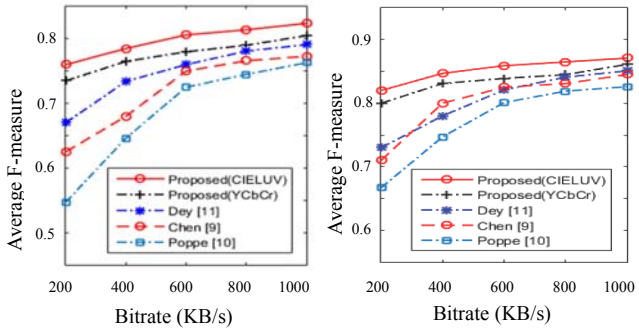
On a final note, the key issue for any successful algorithm is its computational complexity. For the proposed method, we used pre-computed lookup tables to replace any run-time computation associated with F_1, while the maximum number of operations involved with F_2, D for each MB remains constant. Consequently, the running time of the proposed method reduces to O(N), which is *linear* in terms of the number of MBs/frame. It may be noted that background segmentation is but one component of a potentially complex computer vision system. The problem of foreground-background separation is only fundamental, and its results are utilized in conjunction with vision-based problems such as tracking, surveillance and gesture and event detection in real-time. There is, therefore, a huge requirement of high-speed computing techniques as the demand for real-time analysis of multiple surveillance-feeds grows over a constrained-bandwidth network.

V. Conclusion

Background modeling is one of the key problems for automatic video analysis. In this paper, we proposed novel block-based features for dynamic background modeling to work directly on low bit rate encoded videos. Experimentally, the method has been tested on H.264 High profile, a premier codec used for streaming of surveillance-grade high-definition video at constrained bit rates. The method clearly outperforms

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2614984, IEEE Transactions on Circuits and Systems for Video Technology

the current benchmark in terms of segmentation performance, while consuming significantly less CPU time. Considering the bandwidth crunch which mounts a huge challenge to deliver high-quality video, the proposed method is geared to tap into the benefits of superior compression that H.264 has to offer.



(a) Sequence: Fall (b) Sequence: PeopleInShade

Fig. 5. Impact of video encoding bitrates on the segmentation performance of various algorithms.

REFERENCES

[1] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 11-12, pp. 31-66, May. 2014.

[2] 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.

[3] J. Wen, Y. Xu, J. Tang, Y. Zhan, Z. Lai, and X. Guo, "Joint Video Frame Set Division and Low-Rank Decomposition for Background Subtraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 12, pp. 2034-2048, Dec. 2014.

[4] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[5] Recommendation ITU-T H.264 (04/2013): Advanced video coding for generic audiovisual services, ITU-T, Apr. 2013.

[6] "H.264 video compression standard - New possibilities within video surveillance," *White paper,* Axis Communications Inc., Mar. 2008.

[7] W. Zeng, J. Du, W. Gao, and Q.M. Huang, "Robust moving object segmentation on H.264/AVC compressed video using the block-based MRF model," *Real-Time Imaging*, vol. 11, no. 4, pp. 290–299, Aug. 2005.

[8] C. Solana-Cipres, G. Fernandez-Escribano, L. Rodriguez-Benitez, J. Moreno-Garcia, L. Jimenez-Linares, "Real-time moving object segmentation in H.264 compressed domain based on approximate reasoning," *Int. J. Approx. Reasoning*, vol. 51, pp. 99-114, Sep. 2009.

[9] Y.-M. Chen, I. V. Bajic, and P. Saeedi, "Moving region segmentation from compressed video using global motion estimation and Markov random fields," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 421–431, Jun. 2011.

[10] C. Poppe, S.D. Bruyne, T. Paridaens, P. Lambert, R.V.D. Walle, "Moving Object Detection in the H.264/AVC compressed domain for video surveillance applications," *J. Vis. Commun. Image Representation*, vol. 20, no. 6, pp. 428-437, Aug. 2009.

[11] 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.

[12] C. Benedek and T. Szirányi, "Study on Color Space Selection for Detecting Cast Shadows in Video Surveillance," *Int'l J. Imaging Syst. Technol.*, vol. 17, no. 3, pp. 190-201, Oct. 2007.

[13] A. Hallapuro and M. Karczewicz, "Low complexity transform and quantization," Joint Video Team (JVT) Docs. JVT-B038 and JVT-B039, Jan. 2002.

[14] S. Gordon, D. Marpe, and T. Wiegand, "Simplified Use of 8x8 Transforms – Updated Proposal & Results," Doc. JVT-K028, Mar. 2004.

[15] 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.

[16] 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, vo. 11, pp. 4663-4679, Nov. 2014.

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

[18] K. Jack, Video Demystified: a Handbook for the Digital Engineer, LLH Technology Publishing, 3rd ed., 2001.

[19] D. Rogers, Procedural Elements for Computer Graphics, McGraw-Hill, 1985.

[20] ITU-R Recommendation BT.709, Basic Parameter Values for the HDTV Standard for the Studio and International Programme Exchange [formerly CCIR Rec.709] ITU, Geneva, Switzerland, 1990.

[21] F.R. Hampel, "A General Qualitative Definition of Robustness," *The Annals of Mathematical Statistics*, vol. 42, no. 6, pp. 1887-1896, 1971.

[22] G. E. Forsythe, M. A. Malcolm, and C. B. Moler, *Computer Methods for Mathematical Computations*, Prentice-Hall, 1976.

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

[24] M. Paul, W. Lin, C. T. Lau, and B.-S. Lee, "Video coding using the most common frame in scene," in *Proc. IEEE ICASSP*, Mar. 2010, pp. 734–737.

[25] M. Paul, W. Lin, C.-T. Lau, and B.-S. Lee, "A Long-Term Reference Frame for Hierarchical B-Picture-Based Video Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 10, pp. 1729–1742, Oct. 2014.

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 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.