Fast and Efficient Transcoding Based on Low-Complexity Background Modeling and Adaptive Block Classification

Xianguo Zhang, Student Member, IEEE, Tiejun Huang, Senior Member, IEEE, Yonghong Tian, Senior Member, IEEE, Mingchao Geng, Siwei Ma, Member, IEEE, Wen Gao, Fellow, IEEE

Abstract—It is in urgent need to develop fast and efficient transcoding methods so as to remarkably save the storage of surveillance videos and synchronously transmit conference videos over different bandwidths. Towards this end, the special characteristics of these videos, e.g., the relatively static background should be utilized for transcoding. Therefore, we propose a fast and efficient transcoding method (FET) based on background modeling and block classification in this paper. To improve the transcoding efficiency, FET adds the background picture, which is modeled from the originally decoded frames in low complexity, into stream in the form of intra-coded G-picture. And then, FET utilizes the reconstructed G-picture as the long-term reference frame to transcode the following frames. This is mainly because our theoretical analyses show that G-picture can significantly improve the transcoding performance. To reduce the complexity, FET utilizes an adaptive threshold updating model for block classification and then adopts different transcoding strategies for different categories. This is due to the following statistics: after dividing blocks into categories of foreground, background and hybrid ones, different block categories have different distributions of prediction modes, motion vectors and reference frames. Extensive experiments on transcoding high-bit-rate H.264/AVC streams to low-bit-rate ones are carried out to evaluate our FET. Over the traditional full-decoding-and-full-encoding methods, FET can save more than 35% of the transcoding bit-rate with a speed-up ratio of larger than 10 on the surveillance videos to be transcoded more efficiently. On the conference videos which should be transcoded more timely, FET achieves more than 20 times speed-up ratio with 0.2dB gain.

Index Terms—surveillance and conference videos, transcoding, background modeling, classification

I. INTRODUCTION

VIDEO surveillance and video teleconferencing systems are more and more widely used for safety and communication applications. These systems usually adopt common video codecs such as H.264/AVC with common settings to compress captured videos for weeks or months. Compared with general videos, surveillance and conference videos always own much lower coding bit-rate at the same quality. As a result, the deployed common video codecs always compress them at much higher bit-rates. For video surveillance applications, the high-bit-rate video streams greatly enlarge the video storage and retrieval cost. According to the statistics, if the more than 5 million cameras in UK are all High-Definition (HD) ones with general H.264/AVC video codecs, at least 8,000,000 Terabytes data will be produced in one-month storage time. Thereby for surveillance videos, high-efficiency and low-complexity bit-rate scaling techniques are in urgent need to transcode high bit-rate videos to low-bit-rate ones. Moreover, there should be remarkable bit saving compared with the traditional full-decoding-and-full-encoding (FDDE) transcoder. As for video teleconferencing applications, with the exponentially increasing usage of different teleconferencing clients (e.g., mobile devices), the high-bit-rate streams are required to be real-time and simultaneously transcoded into multiple quality-maintained low-bit-rate conference videos for the multiple bandwidths of various client devices. Thereby it is increasingly becoming an important issue to develop faster-than-real-time transcoders to broadcast multiple quality-maintained conference videos to various devices with different bandwidths. Summarized from the requirements of surveillance and conference video transcoding, it is in urgent need to develop specially-designed methods to transcode surveillance videos with large bit saving in low complexity and transcode conference videos to quality-maintained streams as fast as possible.

One intuitive transcoder satisfying the basic requirement is to directly connect common video decoder and encoder. However, it is not very practical due to the transcoding complexity and sometime large degree of transcoding quality loss. To decrease the complexity, many beneficial motion estimation (ME) and mode decision (MD) simplification methods (e.g. [1-7]) have been proposed. But seldom were proposed specially for surveillance and conference videos. Whereas to improve the transcoding quality or efficiency, three methods can be utilized: the object-oriented transcoding methods [8-9] based on object segmentation, the region based methods which employed more bits on regions of interest [10], and the block-based background-prediction optimization methods [11-13]. Although some problems still exist in these methods, they enlightened us utilize better and low-complexity modeled static background to optimize video transcoding for background pixels.
In this paper, to analyze what kind of background can mostly improve the transcoding efficiency, a theoretical comparison using some conclusions in [14] is firstly carried out among three typical background frames, including the key frame in [11], background modeled from reconstructed frames in [12],[13] and the proposed G-picture modeled from the originally decoded frames. The comparison result shows G-picture is the optimal long-term reference frame. In further, a theoretical analysis for how to quantize G-picture in transcoding shows that, G-picture can most significantly improve the rate-distortion (RD) performance while intra-encoded with the minimum quantization parameter (QP) of the input stream. To evaluate the complexity-saving room, another experimental analysis is carried out to utilize G-picture to figure out blocks’ motion characteristics. After blocks are divided into foreground units, background units and hybrid units by calculating their difference the static background, the statistics based on such block classification show: different reference frame candidates should be used for prediction among categories of transcoding units; motion search range should be calculated differently from the difference between decoded motion vector and predicted motion vector; different prediction modes should be available for different categories.

Based on these analysis results, we propose a fast and efficient transcoding (FET) method based on low-complexity background modeling and adaptive block classification. In order to improve the transcoding efficiency, FET utilizes the G-picture, which is online trained by a low-complexity and high-quality background modeling using the originally decoded frames as input, as the long-term reference frames to transcode each decoded frame. Because the G-picture is very clean, encoded only using intra prediction and quantized smaller QP, such a better modeled and encoded G-picture will provide better long-term reference for the following frames. Even for conference videos, in which the background is usually covered by tightly-moved foreground in large areas, the unclean modeled G-picture can also enlarge the transcoding efficiency to some degree. Meantime, to reduce the complexity, FET employs G-picture to realize an adaptive threshold-updating model to achieve adaptive block classification and adopt different transcoding strategies for different block categories. These strategies are in forms of removing redundant prediction modes, simplifying motion estimation and reducing reference frames. Such adaptive block classification reduces the complexity dramatically by employing different ME&MD strategies on different block categories. In summary, as an extension of our work in [15-16], besides [15]’s background model based high-efficiency transcoding and [16]’s block-classification based speed-up strategies, this paper makes improvements on: the theoretical proof for the efficiency of transcoding with properly-quantized G-picture as reference, the low-complexity and high-efficiency background modeling algorithm to generate G-picture, the adaptive-threshold updating based block classification and extensive experiments for both surveillance and conference videos.

To assess the significant bit saving of our FET on surveillance videos and the remarkable complexity saving for both conference videos and surveillance videos, extensive experiments are conducted. These experiments are carried out on eight surveillance videos from AVS (Audio and Video coding Standard) workgroup and eight conference videos from JCT-VC (Joint Collaborative Team on Video Coding). These experiments include the background modeling efficiency, block classification result and the final results for transcoding efficiency improvement and complexity reduction. These results are figured out during transcoding high-bit-rate H.264/AVC streams to low-bit-rate ones. To demonstrate the efficiency, traditional FDFE method directly using H.264/AVC for re-encoding is chosen as the basic anchor.

Such above extensive experiments show that, for surveillance/conference videos, FET averagely saves more than 35%/5% bit saving, equivalent to more than 1.1/0.2 dB PSNR (peak signal-to-noise ratio) gains. Meanwhile, larger than 10/20 and 2/3 times speed up using full search ME and fast ME methods respectively. While compared with the more efficient transcoding with long-term key frame as background reference, the result is also very significant. Moreover, the block-classification based fast method in FET averagely achieves 0.5 times speed up than the method not relying on block-classification, with the similar transcoding quality. To practically qualify our method, two real-time transcoding systems based on FET are designed to respectively transcode HD surveillance videos in much lower bit-rates and HD conference videos to different bit-rates for different bandwidths. In this way, FET is practically proved very efficient.

The rest of this paper is organized as follows. The related works for surveillance and conference video transcoding are discussed in Sec. II, and the theoretical analysis for the efficiency improvement with G-picture is presented in Sec. III. Sec. IV presents the framework and the methods, where the analyses for each block category’s distributions of prediction modes, motion vectors and reference frames are included in the sub-sections for the speed-up methods. Experimental setup is given in Sec. V, and the extensive experimental results are shown in Sec. VI and Sec. VII concludes this paper.

II. RELATED WORKS

Generally, FDFE is the simplest video transcoding approach without any change on the encoding process of decoded videos. However, due to the complexity and efficiency, FDFE is not applicable in practical transcoding systems. For complexity, because the incensement is mainly caused by ME and MD, several fast transcoding methods using motion vector refinement were proposed by [1-4] to decrease ME complexity, with comparable performance to FDFE. Meanwhile, methods for saving MD complexity [5-7] were also widely investigated in the past years. For example, a zero-block decision based scheme was introduced by Wu et al. [5], where the zero-block decision scheme was used to skip impossible inter and intra prediction modes, consequently leading to 93% saving of MD time, on average. Nevertheless, seldom methods specially focused on complexity reduction of surveillance and conference videos. In these videos, blocks with different proportion of foreground pixels have different motion characteristics, so simplifying the transcoding processes in MD and ME strategies for relative static regions will intuitively save the time cost with little quality loss.

For efficiency, because most of the corresponding cameras are mounted to a fixed scene for a long-time and each frame can
be subjectively divided into foreground and background objects, some pioneer works started to employ the static background to improve the efficiency. Intuitively, a reasonable solution following such idea is to transcode foreground objects and background separately. We denote it as object-oriented transcoding throughout this paper. Object-oriented methods were firstly proposed in [8-9] to divide an input frame into foreground and background regions, and then transcode background with low bit-rate. However, object-oriented methods usually focused on subjectively measured “foreground objects.” For surveillance and conference video transcoding, the subjective measurement is a debatable problem, especially considering various security requirements. Besides, the accurate automatic foreground segmentation is still an open problem, so it is also a great challenge to use few bits to compress the object representation information and the foreground prediction residual.

Figure 1. The examples of the "exposed background regions" are shown in the current frame which are divided into foreground and background regions respectively. The circled regions in the "current frame" can only find good reference in the "G-picture" rather than the "key frames" and the "recent reference frames." Usually, there are more such regions in surveillance videos because objects in surveillance videos move more intensely.

To avoid the challenging object segmentation and improve the transcoding efficiency, some efficient block-based coding methods which already utilize the static background characteristics can also be applied to the encoding procedure in transcoding. The methods include the region-based, long-term key frame and background prediction based coding. Among them, the region-based coding [10] mainly focused on achieving high compression efficiency and better subjective quality of foreground regions with low encoding complexity, but the total bit-rate was not decreased very much. The long-term key frame based coding utilized the high-quality encoded key frame as long-term reference [11] for follow-up frames, but there were still some so-called “exposed background” regions that appeared in the current frame and disappeared in the recent reference frames and the key frame. An example for the distributions of the exposed background can be seen in Fig. 1. As seen, there are some circled regions which can find better reference in the G-picture (although the background in conference video is usually not very clean). As a result, the transcoding efficiency for these regions could not be improved by using only the key frame and the recent decoded frame as reference. To address this problem, background prediction based coding methods were proposed in [12] and [13]. Both H.120 in [12] and M. Paul et al. [13] featured at exploiting the reconstructed frames to model the background and employed the background as an additional reference for coding the following frames. However, quality of the generated background could not be guaranteed because significant quantization loss existed in the utilized reconstructed frames. Moreover, high-complexity background generation would be embedded in decoder to guarantee decoding match. Although there were some problems in the optimized methods above, they still enlightened us to improve the efficiency of the "exposed background regions" with better and low-complexity static background as reference.

Following the above ideas for complexity and efficiency, it is very practical to improve the efficiency of exposed background regions using the better modeled G-picture and decrease the complexity according to the motion characteristics of the input blocks. Therefore, we propose to employ the long-term G-picture to facilitate more efficient background prediction and utilize block-classification based speed-up strategies for three categories of blocks.

III. EFFICIENCY ANALYSIS

To begin with our analysis and discussion, the symbols used in this paper are defined in Table 1.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_1$</td>
<td>Displacement error variance with one hypothesis, which usually denotes a reference frame in transcoding.</td>
</tr>
<tr>
<td>$\sigma_x$</td>
<td>Residual noise component that cannot be predicted by motion compensation under a hypothesis.</td>
</tr>
<tr>
<td>$\Phi(\Lambda)$</td>
<td>PSD (power spectral density) of the input 2-D and spatial-frequency signal $\Lambda=(\omega_{x}, \omega_{y})$.</td>
</tr>
<tr>
<td>$\Phi(\Lambda)$</td>
<td>$\Phi(\Lambda)$ denotes the $\Phi(\Lambda)$ with $X$ as long-term reference</td>
</tr>
<tr>
<td>$P(\Lambda)$</td>
<td>PDF (probability density function) of $\sigma_1$. $P(\Lambda)$ reflects the motion compensation accuracy.</td>
</tr>
<tr>
<td>$\Phi_{\sigma}(\Lambda)$</td>
<td>The power spectrum of residual noise component $\sigma_1$.</td>
</tr>
<tr>
<td>$\Phi_{\omega}(\Lambda)$</td>
<td>The non-negative signal power spectrum of the input video signal</td>
</tr>
<tr>
<td>$\Phi_{OB}(\Lambda)$</td>
<td>The power spectrum of residual noise component $\sigma_1$.</td>
</tr>
<tr>
<td>$OB$</td>
<td>Sample matrix of the reconstructed result of the proposed G-picture</td>
</tr>
<tr>
<td>$RB$</td>
<td>Sample matrix of the background trained from the reconstructed result of transcoding each frame.</td>
</tr>
<tr>
<td>$KB$</td>
<td>Sample matrix of the high-quality transcoded key frame</td>
</tr>
</tbody>
</table>

As discussed, the key to improve the transcoding performance is to explore high-quality background data from the decoded frames. Following this, the efficiency of the exposed background regions will be improved with the help of the better long-term background. Although the idea is very straightforward, there is no theoretical analysis so far on what is the optimal background. In this section, we firstly theoretically prove that the G-picture represented by $OB$ is the optimal long-term
reference frame for efficiently transcoding decoded frames. Secondly, we analyze that encoding G-picture into the stream with the minimum decoded QP can guarantee the optimal RD performance for transcoding.

A. Why G-picture is Optimal for Transcoding the Background
As stated, RB is trained from the reconstructed results of the decoded frames, KB is one of the originally decoded frames and OB is the background trained from the originally decoded frames. Therefore, OB could combine the advantages of RB and KB and is probably a better long-term reference frame. In terms of prediction distortion, using OB as the long-term reference frame will achieve less distortion than that using RB or KB for any long surveillance or conference video. More formally, this conjecture can be expressed as Lemma 1.

Lemma 1: Let \( D(\Lambda, OB) / D(RB, \Lambda) / D(KB, \Lambda) \) respectively denote the prediction distortion between a decoded long surveillance or conference sequence \( \Lambda \) and the long-term OB/RB/KB.

Using the same motion search method, the following equation is satisfied in transcoding:

\[
D(OB, \Lambda) \leq \min \{D(KB, \Lambda), D(RB, \Lambda)\}. \tag{1}
\]

Proof: For any transcoding unit at the position of \((x, y)\) in an exposed background region (if any) of an input frame \(I \in \Lambda\), we can find a matched block in OB but there is no such block in KB, RB may contain similar block but its quality is probably poorer than OB due to the quantization loss of the frames used to reconstruct it. Thus on the probability, we have

\[
\left[ (x,y) : D(I(x,y),OB) \leq D(I(x,y),DB), D(I(x,y),KB) \right] \subseteq \left[ (x,y) : I(x,y) \in P \right]
\]

where \(P\) denotes the set of exposed background regions and \(|X|\) is the size of the set \(X\). For surveillance and conference videos, as lots of such regions exist in each decoded frame \(I\), we can get

\[
D(I, OB) = \sum_{x,y} D(I(x,y), OB) \leq \sum_{r,s} \min\{D(I(x,y), KB), D(I(x,y), RB)\}.
\tag{3}
\]

Because

\[
\min\{D(I(x,y), KB), D(I(x,y), RB)\} \geq \sum_{r,s} \min\{D(I(x,y), KB), D(I(x,y), RB)\},
\tag{4}
\]

we can get

\[
D(I, OB) = \sum_{x,y} D(I(x,y), OB) \leq \min\{D(I, RB), D(I, KB)\}.
\tag{5}
\]

Thus for the decoded long sequence, we can obtain

\[
D(\Lambda, OB) = \sum_{i=1}^{d} D(I, OB) \leq \sum_{j=1}^{d} \min\{D(I, RB), D(I, KB)\}
\leq \min\{D(\Lambda, RB), D(\Lambda, KB)\}.
\tag{6}
\]

As stated in [14], for any two reference frames, the one providing smaller prediction distortion and smaller \(\Phi(\Lambda)\) will lead to a better rate-distortion performance. The distortion relationship has been discussed in lemma 1. To regard the \(\Phi(\Lambda)\)s of OB, RB and KB, we have Lemma 2.

Lemma 2: Let \(\Phi_{OB}(\Lambda), \Phi_{RB}(\Lambda)\) and \(\Phi_{KB}(\Lambda)\) denote the PSDs of a decoded surveillance or conference sequence with OB, RB and KB as the long-term reference frames respectively. Using the same motion search method, the following equation is satisfied in surveillance and conference video transcoding:

\[
\Phi_{KB}(\Lambda) < \min\{\Phi_{PB}(\Lambda), \Phi_{RB}(\Lambda)\}.
\tag{7}
\]

Proof of Lemma 2 is given in the Appendix. By combining Lemma 1 and 2, we can get Theorem 1.

Theorem 1: Let \(RD(OB, \Lambda) / RD(RB, \Lambda) / RD(KB, \Lambda)\) denote the rate-distortion performance between a decoded surveillance or conference sequence \(\Lambda\) and OB/RB/KB. Using the same motion search on the long-term reference frames OB, KB and RB, the following equation is satisfied in transcoding:

\[
RD(OB, \Lambda) < \min\{RD(KB, \Lambda), RD(RB, \Lambda)\}. \tag{8}
\]

Proof: Again as stated in [14], between any two prediction reference frames, if using one frame as reference can obtain smaller prediction distortion and smaller \(\Phi(\Lambda)\), the reference frame can help to achieve a better RD result. Because \(D(OB, \Lambda)\) is proved in Lemma 1 to be the minimum among \(D(X, \Lambda)\)s and \(\Phi_{OB}(\Lambda)\) is also derived in Lemma 2 to be the minimum among \(\Phi_{X}(\Lambda)\)s, \(RD(OB, \Lambda)\) is also minimum.

In summary, by utilizing OB as the long-term reference frame, the transcoding efficiency of the decoded frames in surveillance and conference videos will be significantly improved. As stated in [14], the less prediction error variance (PEV) leads to less \(\Phi_{OB}(\Lambda)\), so using the long-term OB with less \(\Phi_{OB}(\Lambda)\) might produce less PEV. To validate this, we experimentally calculate the average PEV between each input frame and the long-term OB/KB/RB. Fig. 2 shows the results for two sequences, crossroad (352x288) and overbridge (352x288). We can see that, after several initial frames, PEV for OB becomes less than that of KB and RB. Moreover, the gap between OB and KB/RB becomes larger and larger as frame number increases. This is because OB contains more higher-quality background pixels and less noise or foreground pixels.

B. How to Quantize G-picture for the Least RD Cost
For the decoding match of using G-picture as the long-term reference frame in surveillance and conference video transcoding, we should encode G-picture into stream. Thus another problem is how large should be the QP for quantizing it? As stated in [23], the Lagrange RDO theory calculates the Lagrange cost for each sequence from the Lagrange cost of each frame by

\[
J = \sum_{i=1}^{d} RD(I, J, P_{J}, V_{J}), \tag{9}
\]

where \(n\) is the total number of frames in the sequence, \(I_{J}\) is the \(k\)-th transcoding unit of the \(j\)-th frame \(I_{J}, V_{J} \) is the prediction motion vector, \(P_{J}\) denotes the predicted data of the current transcoding unit and \(RD(Q, I_{J}, P_{J}, V_{J})\) is the Lagrange cost for coding \(I_{J}\) with \(Q\) as the QP. Particularly, while coding with G-picture as the long-term reference frame, \(P_{J}\) can be described by

\[
P_{J} = \Phi(I_{J}, OB, Q_{J}, R_{J}), \tag{10}
\]

where \(Q_{J}\) is the QP for coding OB and \(R_{J}\) is \(I_{J}\)’s set of reference frames excluding the long-term reference frame. Because G-picture is not an original input frame, Eq. 9 turns to be

\[
J(Q_{J}) = \lambda_{J}(Q_{J}) \times R_{J}(Q_{J}, OB) + \sum_{i=1}^{d} RD(Q, I_{J}, \Phi(I_{J}, OB, Q_{J}, R_{J}), V_{J}), \tag{11}
\]
where function $R_g$ calculates the bit cost for coding the OB with QP equal to $Q_g$, and $\lambda_g$ is the Lagrange multiplier of $R_g$. As how Theorem 1 is derived, any prediction reference $X'$ which provides less sufficient reference (i.e., larger prediction distortion) than $X$ will lead to

$$RD(Q, I_{j,k}, X', V_{j,k}) \geq RD(Q, I_{j,k}, X, V_{j,k}).$$

(12)

Because the larger QP produces larger distortion, for any positive integer $q$, we further have

$$RD(Q, I_{j,k}, \Omega(I_{j,k}, Q_g, R_j), V_{j,k}) \geq RD(Q, I_{j,k}, \Omega(I_{j,k}, Q_g, R_j, Q_g, OB), V_{j,k}),$$

(13)

where $RD(Q_g) = RD(Q, Q_g) = A(Q_g, q) - \sum_j B_j(Q_g, q)$.

$$A(Q_g, q) = (\lambda_g Q_g + R_g Q_g) \Omega(I_{j,k}, OB) - R_g Q_g, OB, Q_g, OB),$$

(14)

$$B_j(Q_g, q) = RD(Q_g, I_{j,k}, \Omega(I_{j,k}, OB, Q_g, R_j, V_{j,k}) - RD(Q_g, I_{j,k}, \Omega(I_{j,k}, OB, R_j, V_{j,k})).$$

(15)

Note that, the $A(Q_g, q)$ in this equation is not less than zero because positive cost the intra-coded OB will turn smaller with a larger QP, and $B(Q_g, q)$ is also larger than zero because of Eq. 13. Moreover, supposing $Q_D$ is the minimum QP of the decoded QPs from the input stream, we can derive

$$Q_g + R_g Q_g, q \Omega(I_{j,k}, OB) - R_g Q_g, OB, Q_g, OB),$$

(16)

This is because OB is trained from the original decoded frames which already had the $QP^D$-level quality loss, and quantizing OB could not make the quality loss less than $QP^D$-level. Therefore, we can get

$$J(Q_g) - J(Q_g + q) = \begin{cases} A(Q_g, q), & Q_g + q \leq Q_D \\ A(Q_g, q) - \sum_j B_j(Q_g, q), & Otherwise \end{cases}.$$ 

(16)

In Eq. 16, because surveillance and conference videos always capture the same scene for long-time, one G-picture can long-term predict large number of following. That means, $n$ is very large and

$$J(Q_g) - J(Q_g + q) = \begin{cases} A(Q_g, q), & Q_g + q \leq Q_D \\ J(Q_g) - J(Q_g + q) = A(Q_g, q) - \sum_j B_j(Q_g, q), & Otherwise \end{cases}.$$ 

(17)

This means, $Q_g$ is the best QP to quantize OB and achieve the minimum total rate-distortion cost.

$$J(Q_g) = \min J(Q_g), Q_g \text{ ranges from the smallest QP to the largest}.$$ 

(18)

Thus in our FET, G-picture should be quantized with the minimum decoded QP. To verify the theory, we have employed different QPs for G-picture to obtain the total bits and the transcoding PSNR for an input stream. With an input stream of crossroad (CIF, 352x288) encoded it with QP=17, the coding bit-rate and PSNR curves utilizing the long-term OB quantized with QP=11-24 can be seen from Fig. 3. As is seen, using QP=17 to quantize OB leads to the least bit cost and best PSNR.

Besides the necessary of utilizing the long-term and properly-quantized G-picture improve the transcoding efficiency, a low-complexity and high-efficiency background modeling algorithm should be embedded into FET to generate a high-quality G-picture. As for reducing the transcoding complexity, because the classified transcoding units, including background unit (BU), foreground unit (FU) and hybrid unit (FBU), always have different motion characteristics, FET should employ different MD and ME strategies for each category. Therefore, the generalized framework of the proposed FET is constructed as shown in Fig. 4. It works as follows: firstly, a background frame is generated from the originally decoded frames by Low-complexity Background Modeling, and then this background should be encoded into stream by Background Encoding. The reconstructed result of Background Encoding is used as a selective long-term reference frame for following decoded frames. After that, the Adaptive Block Classification utilizes an adaptive updated and auto learning threshold to divide the blocks in current frame into FUs, FBUs and FBUs. Thirdly, with the help of decoded data (i.e., reference frames, motion vectors and prediction modes), different ME and MD strategies (i.e., Reference Frame Selection, Candidate Modes Calculation and Motion Estimation Intensity Evaluation) will be respectively used for the three block categories.

In the following parts of this section, Sec. IV-A introduces the low-complexity and high-efficiency background modeling algorithm used in the Low-complexity Background Modeling; Sec. IV-B presents the algorithm of block classification based on threshold updating for the Adaptive Block Classification; The complexity analyses and summarized methods respectively for Reference Frame Selection, Candidate Modes Calculation and Motion Estimation Intensity Evaluation are introduced in Sec. IV-C, IV-D and IV-E. In these complexity analyses, an H.264/AVC-transcoding is used to derive the distributions of the optimal reference frames, best prediction modes and motion vectors for BUs, FUs and FBUs, and speed-up strategies are respectively summarized for them. The experiments are conducted on four representative surveillance and conference videos (surveillance videos of crossroad/overbridge and conference videos of mthr_dor/paris, all of which can be seen from Sec. V), whose input H.264/AVC stream for transcoding is at about 1000 kbps. These videos are more representative because they contain different characteristics of bright/dark scenes, large/small moving objects and fast/slow motions.

Figure 3. The transcoding PSNR and bit-rates with different QPs for G-picture, where the minimum decoded QP=17 helps achieve the best bit-rate and PSNR.

Figure 4. Framework of the Proposed FET.
To maintain or improve background quality, an ideal solution for background modeling is to calculate the mean value of all the purely background pixels in the training frames. However, it is very difficult in recent years to exactly justify which pixels belong to the background. Physically, “background” equals to the most frequently-appearing content. This inspires FET to utilize a novel segment-and-weight based running average (SWRA) to approximately calculate background by paying larger weight on the frequently-appearing values in the averaging process. Because SWRA is based on a running average procedure, there will not be large memory cost and computational complexity. Generally, SWRA divides the pixels at a position in the training frames into temporal segments with their own mean values and weights, and then calculates the running and weighted average result on the mean values of the segments. In the process, pixels in the same segment have the same background/foreground property and the long segments have larger weight. This method ignores the foreground/background property of each segment, so foreground recognition is avoided. Meanwhile, low memory cost and no-delay modeling are guaranteed.

In detail, SWRA models a background value of pixels at position \((x, y)\) by following five steps:

1. **Initialize.** Initialize background model value \(AVG\) and its weight \(W\) for the following weighted average procedure to 0, and then create first segment. Length of the first segment \(L\) equals to 0 and its mean value \(avg=0\). The model value before the current segment \(avg'\) is also set 0.

2. **Calculate the threshold for segmenting.** Supposing \(\mu\) is the mean value, \(\sigma^2\) is the mean square error, the probability of \(|f(x) - \mu| > 2\sigma\) in normal distribution \(f(x)\) is less than 4\%. So we use 2\(\sigma\) as the threshold \(th\) to temporally segment a pixel in training frames. The threshold \(th\) is initialized to 14 and updated by 2 times the root square value of the mean of gap values not larger than the \(th\) before. The gap value is the difference between a pixel and its \(avg\).

3. **Create a new segment or widen the current segment.** At arbitrary position, a new temporal segment will be created if \(|f(x, y) - AVG_{i-1}| > th\) is larger than \(th\). Otherwise, length of the current segment is widened. Through this procedure, temporally successive pixels can be divided into segments as shown in Fig. 5. Borders between segments stand for a texture switch on adjacent frames. Note that, if length of a segment is too short, the weight of for the segment is 0, and 1/20 of the length of training frames is used to judge whether a segment is too short.

4. **Calculate mean value and weight for each segment.** The weight of each segment is set square of its length, as shown in Fig. 5. Afterwards, denoting length and mean value of segment \(k\) as \(len_k\) and \(avg_k\), a running average procedure will be employed to realize low computational complexity.

5. **Generate and output the background value.** In a practical system, to satisfy low memory cost, we do not buffer the length and mean values of each segment. Instead, we just interactively

### Table 2. Memory cost for each pixel (byte)

<table>
<thead>
<tr>
<th>ITEM</th>
<th>RA</th>
<th>GMM-1</th>
<th>GMM-5</th>
<th>MS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffered frames</td>
<td>1× (char)</td>
<td>1× (char)</td>
<td>1× (char)</td>
<td>1× (char)</td>
</tr>
<tr>
<td>Mean values</td>
<td>2× (double)</td>
<td>2× (double)</td>
<td>2× (double)</td>
<td>M× (char)</td>
</tr>
<tr>
<td>Weight</td>
<td>0</td>
<td>1× (double)</td>
<td>1× (double)</td>
<td>0</td>
</tr>
<tr>
<td>Threshold/variace</td>
<td>0</td>
<td>1× (double)</td>
<td>1× (double)</td>
<td>0</td>
</tr>
<tr>
<td>Match point number</td>
<td>0</td>
<td>1× (char)</td>
<td>1× (char)</td>
<td>0</td>
</tr>
<tr>
<td>SUM of MEMORY</td>
<td>5</td>
<td>34</td>
<td>34×5=170</td>
<td>M=120</td>
</tr>
</tbody>
</table>

### Table 3. The PSNR gain (dB) from background modeling and the modeling time (second)

<table>
<thead>
<tr>
<th>ITEM</th>
<th>RA</th>
<th>GMM-1</th>
<th>GMM-5</th>
<th>MS</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-BP-x</td>
<td>crossroad</td>
<td>overbridge</td>
<td>snowgate</td>
<td>snowroad</td>
</tr>
<tr>
<td>Gain</td>
<td>Time</td>
<td>Gain</td>
<td>Time</td>
<td>Gain</td>
</tr>
<tr>
<td>GMM-1</td>
<td>0.79</td>
<td>5.9</td>
<td>0.50</td>
<td>5.9</td>
</tr>
<tr>
<td>RA</td>
<td>0.93</td>
<td>1.6</td>
<td>0.80</td>
<td>1.7</td>
</tr>
<tr>
<td>MS</td>
<td>1.01</td>
<td>11.5</td>
<td>0.89</td>
<td>11.3</td>
</tr>
<tr>
<td>GMM-5</td>
<td>1.22</td>
<td>61.8</td>
<td>0.95</td>
<td>61.2</td>
</tr>
</tbody>
</table>

### A. Background Modeling

Recently, background modeling has been utilized for efficient surveillance video coding and transcoding. In this section, we will firstly analyze and compare among existing off-used background modeling methods, and then a background modeling method with low memory cost and computational complexity is proposed to generate G-picture for video transcoding. To evaluate the efficiency of different common background modeling methods for video transcoding, four typical background modeling methods are implemented and embedded into the FDFE method with H.264/AVC baseline profile. They transcode the input H.264/AVC streams at 1000 kbps for four sequences (crossroad, overbridge, snowgate and snowroad) to the output streams at bit-rates of 64, 128, 256 and 512 kbps. The four methods are the Gaussian Mixed Models [17] using 1 or 5 models for each pixel (GMM-1 or GMM-5), the Mean-Shift (namely MS) proposed in [18], and the popularly used Gaussian running average (RA). For background modeling in surveillance and conference video transcoding, as is referred in Picardi’s [19], performance, memory cost and running time are the same important factors. The calculations for their memory cost in each pixel position are listed as follows. (1) RA: one current pixel with type of char and one float-precision mean value for each pixel should be buffered. (2) GMM-X: besides the buffered input pixel, a GMM model is required to be buffered. The model is composed of double-precision mean value, variance and weight. Moreover, an 8-bit value should be stored to count the number of matched points for each GMM model. (3) MS: Mean-shift based algorithms usually buffer all the training frames and very few additional temporal variables are used for the clustering and sorting operations.

Supposing the number of training frames is \(M=120\), the memory cost for each algorithm derived from the above analysis is listed in Table 2. Then we implement transcoding methods respectively utilizing RA, MS, GMM-1 and GMM-5 to train G-picture as long-term reference on H.264/AVC baseline profile. The methods are correspondingly named T-BP-MS, T-BP-RA, T-BP-GMM-1 and T-BP-GMM-5, and the transcoding time and efficiency on different CIF sequences can be seen from Table 3. In a brief summary, GMM-5 contributes largest to video coding performance gain but spares a relative large memory and time cost. In practical system, especially in parallelism or hardware environment, such GMM-5 cannot meet the requirement for fast modeling and low memory cost. This inspires us to propose a method which can achieve higher performance with less memory and time cost.

![Figure 5. Calculate the mean value and weight for each segment.](image-url)
buffer and calculate the total mean value \( AVG \) and its weight \( W \) from the first to the \( k \)-th segment by

\[
AVG = \frac{(AVG \times W + len_k^2 \times avg_k)}{(W + len_k^2)}, \quad W = W + len_k^2.
\]

Such calculation procedure is shown in Fig. 6. It indicates we only need to buffer and derive the \( AVG \) and \( W \) of the first \( k \) segments from the first \( k-1 \) segments. Following this, when the current segment reaches the end of training frames, we will calculate the final \( AVG \) and \( W \). At last, we will obtain the required background by jointing the \( AVG \) of each pixel together.

![Fig. 6. The calculation of buffered AVG and W.](image)

From the above statement, we can see that the proposed SWRA works based on weights and running average. The required additional buffered data for each pixel position include: the \( avg_k/len_k \) for the current segment \( k \), the \( AVG/W \) to summarize the previous segments and the updating threshold. Compared with the parametric methods like GMM, SWRA does not import multiple models for each pixel and never relies on the float precision calculation of proportion and variance, so both memory and time are saved; Compared with Mean-Shift, SWRA does not need to allocate large memory to buffer multiple training frames, so memory will be significantly reduced; This method is also different from non-parametric methods like codebook[20], although the codebook does not need to buffer multiple training frames, the management of the multiple codewords is very time consuming and memory sparing. In Sec. V, we will practically count the efficiency improvement and memory-and-time cost of SWRA.

B. Adaptive Block Classification

As discussed above, FET employs different transcoding strategies for different categories of transcoding units. Therefore, a low-complexity and scene-adaptive classification algorithm should be designed to classify units into BUs, FBUs and FUs. In our practice, an adaptive threshold \( Th_4 \) is learned for each transcoding unit to judge the category \( S \). Following the idea that different units have different proportions of foreground, given the \( Th_4 \), \( S \) is calculated by

\[
S = \begin{cases} 
FU, & \left\| (x,y) \right\| |C(x,y)−B(x,y)|<Th_4, 0 \leq x, y < w \right\| w^2 < \alpha \\
FBU, & \left\| (x,y) \right\| |C(x,y)−B(x,y)|<Th_4, 0 \leq x, y < w \right\| w^2 < \beta \\
BU, & \left\| (x,y) \right\| |C(x,y)−B(x,y)|<Th_4, 0 \leq x, y < w \right\| w^2 \geq \beta 
\end{cases}
\]

where \((x,y)\) is the pixel position in the current transcoding unit \( C \), \( B \) is the reconstructed result of transcoding \( OB \), \(|A|\) is the number of elements in set \( A \) and \( w \) is the width of \( C \). In practice, we usually set \( \alpha=5/64 \) and \( \beta=50/64 \). The remaining problem is how to adaptively calculate the threshold \( Th_4 \). To identify the foreground pixels in a new frame, a reasonable idea is to calculate a separate \( Th_4 \) for each unit with help of the root-mean-square deviation \( \sigma \). Following this, we propose an adaptive learning and updating algorithm as shown in Algorithm 1. The threshold calculating process for each unit can be divided into four steps: (1) Calculate the difference between the current unit and its background; (2) Utilize the threshold for the unit in the last frame to identify background pixels in the current unit; (3) Count the number of identified background pixels in the current unit; (4) Calculate the root-mean-square deviation value to update \( Th_4 \).

- **Input:**
  \( I(m,n) \): the pixel value at position \((m,n)\) of the current \( w \times w \) decoding unit in the current frame.
  \( Bg(m,n) \): the background pixel corresponding to the \( I(m,n) \).

**Initialization:**

\( Th_4 \) is initialized as the \( Th_4 \) for co-located coding unit in the previous frame, or 14 for the first frame.

**Calculation:**

1. For each \( 0 \leq m, n \leq w \), calculate \( Diff(m,n) = |I(m,n) − B(m,n)| \).
2. For each \( (m,n) \) position, calculate
   \[ Cmp(m,n) = \begin{cases} 1, & Diff(m,n) \leq 2 \times Th_4 \\ 0, & Diff(m,n) > 2 \times Th_4 \end{cases} \]
3. Count the potential background pixel number by
   \[ Sum = \sum_{m,n} (Cmp(m,n)), \quad 0 \leq m, n \leq w \]
4. Calculate the root-mean-square deviation as the updated \( Th_4 \) for the current coding unit
   \[ Th_4 = 2 \times \text{Round} \left( \frac{\sum_{m,n} (Cmp(m,n) \times Diff^2(m,n))}{Sum} \right) \]

where \( \text{Round}(A) \) denotes the round value of \( A \).

**Output:** \( Th_4 \).

C. Reference Frame Selection

To clearly and objectively analyze the distribution of the selected reference frames for different categories, the number of reference frames is set to 5 in experiments and G-picture is treated as the long-term reference frame. Respectively for BUs, FUs and FBUs, the percentage of one frame being selected as reference is calculated from the selected times of each reference frame for each category of units. Firstly as Fig. 7 shows, the first reference frame takes up more than 30%/50% for all the categories in surveillance/conference videos; the long-term G-picture takes up more than 40%/18% to predict the BUs, and more than 5% to predict FBUs. Secondly, the first two reference frames take up more than 90% to predict FUs; the first and G-picture can take 90% for BUs; the first two and G-picture together take up about 90% in BUs or FBUs.

From the statistics, we can conclude such rule for speeding up reference frame selection: only the first two reference frames are indispensable to FUs; whereas the first reference and the long-term G-picture can together provide sufficient reference for transcoding BUs; while adding the second reference, FBUs can be provided sufficient reference. Moreover, to avoid exceptional cases, the decoded reference frame of current unit should also be utilized for FBUs and FUs. From this rule, the simplified candidate reference frame pool is shown as Table 4.

This means following selection mechanism: For BUs, only G-picture and the nearest reference frame should be added into the candidate reference frame set; For FBUs, the nearest, second nearest and G-picture should be used; For FUs, we should utilize the two nearest reference frames and the decoded reference frames. Due to the decrease of candidate reference frames in BUs/FUs/FBUs, the redundant computation in ME can be obviously reduced.
(1) calculate each prediction unit PMVD(i, j) of the j-th prediction unit of the total number p in the encoder and their corresponding decoded motion vector MVdec(i, j) from the decoder in the transcoding process. However in the decoding process of the i-th unit, the decoded number of MVdec(i, j) for the prediction units is not p. Supposing there are k decoded motion vectors, we utilize PMVD(i, j) to represent the largest value between PMVD(i, j) and MVdec(i, j, k). That means, to maximum the motion estimation accuracy, we figure out the PMVD(i, j) by

\[
PMVD(i, j, x) = \max \{ PMV(i, j, x) - MV_{dec}(i, m, y), m = 1 \sim k \}
\]

(22)

\[
PMVD(i, j, y) = \max \{ PMV(i, j, y) - MV_{dec}(i, m, y), m = 1 \sim k \}
\]

(23)

where PMV(i, j, t) and MVdec(i, j, t) are the motion vector value of PMV(i, j) and MVdec(i, j) in t coordinate.

According to the summarized rule, it will be enough to just employ sub-pixel motion estimation for FBUs, and we should investigate on the search range calculation of FBUs and FUs. To generate a motion search range reducing the accuracy of motion estimation in the least degree, it is intuitive that the search range in X and Y direction must be larger than the minimum value of all the PMVD(i, 1, x) ~ PMVD(i, p, x) and PMVD(i, 1, y) ~ PMVD(i, p, y). Therefore, we calculate the search range Rt(1) ~ Rt(p) for the total p prediction units in t coordinate by following algorithm in Algorithm 2. This algorithm can be summarized by 4 steps: (1) calculate each prediction unit’s category in {FU, FBU, FBU}; (2) Fix every BU’s search range R1 to 1; (3) Set the search range of the FBU to be d1 larger than the prediction unit P(j)’s PMVD(i, j, t); (4) Set the search range of the FU to be d2 larger than P(j)’s PMVD(i, j, t). Take a w×h/2 prediction unit E as example, the search range (Rs, Rs) is shown in Fig. 9.

**Algorithm 2. Search Range Calculating Algorithm.**

**Input value:**

Rs = search range (SR) for the original FDFE; d1/d2: the extra SR for FBU/FU

**Init value:**

Rs = Rs0, prediction unit j is namely P(j), and d1 and d2 are usually set to 2

**Calculation procedure:**

For j=1~k

\[
S(j) = \begin{cases} 
FU, & \text{if } \frac{\left| (x, y) - (X, y) \right|}{\text{Size of } P(j)} < \alpha \\
FBU, & \text{if } \frac{\left| (x, y) - (X, y) \right|}{\text{Size of } P(j)} \geq \beta \\
BU, & \text{else } 
\end{cases}
\]

If S(j)=FU, Then R(j)=1;
Else Begin

If S(j)=FBU, then Flag = 0; Else, Flag=1;
If (PMVD(j, j)=0), then Rmod(j)=1 + \times Flag;
Else if (PMVD(j, j)=1), then Rmod(j)=d1 + d2 \times Flag;
Else if (PMVDmod(j, j)=Rmod(j)), then R(j)=PMVDmod(j, j)\times d1 + d2 \times Flag;
Else R(j)=Rmod;

End

**Output value:** R(1)~R(k)

**Table 4. The selected reference frame for each block category**

<table>
<thead>
<tr>
<th>Candidate reference frame</th>
<th>BUs</th>
<th>FUs</th>
<th>FBUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>First, G-picture</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First, Second, the</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>predicted Reference frame</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First, Second, G-picture</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>the decoded Reference</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**D. Motion Estimation Intensity Evaluation**

To avoid performance loss in video transcoding, motion search range for each unit should be larger than “Real MVD”. Here the so-called Real MVD means the difference between the predicted motion vector (PMV) from the neighboring units and the best matched motion vector. Fig. 8 shows the distribution of the Real MVDS for BUs, FUs and FBUs. As Fig. 8 shows, more than 99% of the Real MVDS are less than 1 integer pixel in BUs, so the transcoding integer motion search range can be set to 1. For FUs and FBUs, although the range of larger Real MVDS does not turn much larger (e.g., more than 10% Real MVDS is larger than 1 pixel), the increased proportion cannot be neglected because the transcoding bit-rate is more easily influenced by these larger Real MVDS.

From the statistics, we can conclude the rule for motion estimation intensity evaluation: in BUs, motion vector is close to the predicted motion vector; the motion search range should be a non-square window based on the difference between the predicted motion vector and the decoded MV; the search window should be narrowed in different degrees for FUs and FBUs. In this paper, for the i-th transcoding unit, denoting PMVD(i, j) as the difference between predicted motion vector PMV(i, j) of the j-th prediction unit of the total number p in the encoder and their corresponding decoded motion vector MVdec(i, j) from the decoder in the transcoding process. However in the decoding process of the i-th unit, the decoded number of MVdec(i, j) for the prediction units is not p. Supposing there are k decoded motion vectors, we utilize PMVD(i, j) to represent the largest value between PMVD(i, j) and MVdec(i, j, k). That means, to maximum the motion estimation accuracy, we figure out the PMVD(i, j) by

\[
PMVD(i, j, x) = \max \{ PMV(i, j, x) - MV_{dec}(i, m, y), m = 1 \sim k \}
\]

(22)

\[
PMVD(i, j, y) = \max \{ PMV(i, j, y) - MV_{dec}(i, m, y), m = 1 \sim k \}
\]

(23)

where PMV(i, j, t) and MVdec(i, j, t) are the motion vector value of PMV(i, j) and MVdec(i, j) in t coordinate.

According to the summarized rule, it will be enough to just employ sub-pixel motion estimation for FBUs, and we should investigate on the search range calculation of FBUs and FUs. To generate a motion search range reducing the accuracy of motion estimation in the least degree, it is intuitive that the search range in X and Y direction must be larger than the minimum value of all the PMVD(i, 1, x) ~ PMVD(i, p, x) and PMVD(i, 1, y) ~ PMVD(i, p, y). Therefore, we calculate the search range Rt(1) ~ Rt(p) for the total p prediction units in t coordinate by following algorithm in Algorithm 2. This algorithm can be summarized by 4 steps: (1) calculate each prediction unit’s category in {FU, FBU, FBU}; (2) Fix every BU’s search range R1 to 1; (3) Set the search range of the FBU to be d1 larger than the prediction unit P(j)’s PMVD(i, j, t); (4) Set the search range of the FU to be d2 larger than P(j)’s PMVD(i, j, t). Take a w×h/2 prediction unit E as example, the search range (Rs, Rs) is shown in Fig. 9.

**Input value:**

Rs = search range (SR) for the original FDFE; d1/d2: the extra SR for FBU/FU

**Init value:**

Rs = Rs0, prediction unit j is namely P(j), and d1 and d2 are usually set to 2

**Calculation procedure:**

For j=1~k

\[
S(j) = \begin{cases} 
FU, & \text{if } \frac{\left| (x, y) - (X, y) \right|}{\text{Size of } P(j)} < \alpha \\
FBU, & \text{if } \frac{\left| (x, y) - (X, y) \right|}{\text{Size of } P(j)} \geq \beta \\
BU, & \text{else } 
\end{cases}
\]

If S(j)=FU, Then R(j)=1;
Else Begin

If S(j)=FBU, then Flag = 0; Else, Flag=1;
If (PMVD(j, j)=0), then Rmod(j)=1 + \times Flag;
Else if (PMVD(j, j)=1), then Rmod(j)=d1 + d2 \times Flag;
Else if (PMVDmod(j, j)=Rmod(j)), then R(j)=PMVDmod(j, j)\times d1 + d2 \times Flag;
Else R(j)=Rmod;

End

**Output value:** R(1)~R(k)
E. Candidate Modes Calculation

It is clearly that the used intra- and inter-prediction modes are entirely different among BUs, FUs and FBUs. Thereby the used prediction modes in transcoding units counted to figure out the proportion of each prediction mode. As Fig. 10 shows, SKIP and inter 16×16 prediction modes are selected almost 100% in BUs. Therefore, the intra, small and non-square modes are forbidden in BUs. For FUs and FBUs, however, although the small modes (8×8, 8×4, 4×8 and 4×4) are not used very much, there is still over 10% for them, on average. Another interesting discovery is that the Intra 16×16 (I16M) prediction mode is barely used in FBUs, because the flat I16M will produce large distortion for the background-and-foreground hybrid blocks.

These distributions indicate the rule for candidate mode calculation: the used modes in surveillance and conference video transcoding are sharply varied in background and foreground units; the large and square inter-prediction modes are efficient enough for transcoding the BUs; the small-size modes for FBUs should be removed only when the decoded unit does not use any of them; the small-size modes for FUs should never be removed. According to this rule, for static regions, the large size prediction modes will be mostly selected, so smaller and non-square prediction modes like inter and intra 4×4(P4×4, 14M) and 8×8(P8×8, 18M), inter 4×8 and 8×4(P4×8, P4×8), inter 8×16 and 16×8(P8×16, P16×8) in H.264/AVC are forbidden in BUs. But these smaller inter modes should be enabled for FUs. The difference between FBUs and FUs is that, large intra prediction mode such as I16M in H.264/AVC should be forbidden in BUs. Therefore, the final mode decision refinement for H.264/AVC transcoding is clearly listed in Table 5, where S denotes the lowest size of decode mode is equal or greater than 8×8 block size. As shown, the candidate prediction mode pool contains three levels, and each level has various sizes of modes.

<table>
<thead>
<tr>
<th>Decode Mode</th>
<th>FU</th>
<th>BU</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>level1, 116M</td>
<td>level2</td>
</tr>
<tr>
<td>P8×4, P4×8</td>
<td>level2, 116M</td>
<td>level3</td>
</tr>
<tr>
<td>P4×4, 14M</td>
<td>level3, 116M</td>
<td>level3</td>
</tr>
</tbody>
</table>

* S: Decode Mode Size={, SKIP, P16×16, P16×8, P8×16, P8×8, 116M}, level1={SKIP, P16×16, P16×8, P8×16, P8×8, 14M}, level2={level1 ∪ {P8×4, P4×8}}, level3={level2 ∪ {P4×4}}

V. EXPERIMENTAL SETUP

A. Methodology

Figure 11. Example frames of tested surveillance sequences.

To evaluate the effectiveness and efficiency of the proposed FET, which transcodes high bit-rate input streams to low bit-rate streams in high efficiency and low complexity, extensive experiments are carried out on different kinds of surveillance and conference videos. For surveillance video, eight long ones are used, including four sequences (crossroad, overbridge, office and bank) in SD definition and four ones (crossroad, overbridge, snowroad and snowgate) in CIF. They cover different scenes including bright and dusky lightness (BR/DU), large and small foreground (LF/SF), fast and slow motion (FM/SM). As shown in Fig. 11, crossroad (SD), overbridge (SD), office (SD) and crossroad (CIF) are brighter than others. Whereas in crossroad (SD), overbridge (SD), office (SD) and crossroad (CIF) and overbridge(CIF), the foreground motion is very fast and the proportion of foreground pixels is relatively large. For conference video, eight JCT-VC videos including two CIF sequences(paris, mthr dotr) and six 720p videos(vidyo1, vidyo3, vidyo4, johnny, KristenAndSara, FourPeople) are utilized to evaluate FET’s efficiency and complexity. These videos can be seen from Fig. 12.

Note that, to calculate the efficiency of FET at different lower bit-rates, the input streams for all the sixteen videos above should be at high bit-rate, so these streams to be transcoded are all compressed by H.264/AVC High Profile using recommended configurations [21] with QP=17. Besides, because low-delay characteristic is required for surveillance and conference video transcoding, an IPPP sequence structure without B frames is utilized. Moreover, the lower bit-rates refer to the following QPs: the eight surveillance videos are with QP=22, 27, 32 and 37; the eight conference videos are with larger QP at 22, 24, 27 and 30 because the compression ratio of conference videos at similar QPs will be too large. All the methods are designed to transcode streams within H.264/AVC standard from higher bit-rates to lower bit-rates, since inside-standard transcoding will be more practical and import fewer problems for stream displaying and communication.
For an undisputed comparison, such above input streams are transcoded by following five high efficient or fast methods with the comparison tool of BD-PSNR in [22]: 1) T-AVC: In the encoding process of the transcoding procedure, it combines the decoder and encoder in the original H.264/AVC high profile of H.264/AVC test model JM17.2, which is configured as [21]. 2) T-KB: It transcodes with the high-quality key frame as the long-term reference frame. 3) FET-E: It is the FET with only the efficiency-improving techniques, that is, using proposed SWRA-based-modeling G-picture as the long-term reference frame, where G-picture is encoded by the minimum decoded QP. 4) FET-EF: It is the FET-E speeded up by the adaptive-block-classification based reference frame selection, candidate modes calculation and motion estimation intensity evaluation. 5) FET-ES: Based on FET-E, it only employs state-of-the-art fast transcoding methods to save MD and ME complexity, in forms of similar but block-classification independent speed-up strategies in FET-EF.

Through the comparison between FET-E and T-AVC/T-KB, we can figure out the transcoding performance gain or bit-rate-saving over the traditional DFDE and optimized DFDE. In further, by comparing between FET-EF and T-AVC/FET-ES, we can calculate the complexity saving of our proposed speed-up techniques with FET-E and the state-of-the-art methods. The common H.264/AVC test model JM17.2 for the transcoders is configured as Table 6.

**Table 6. Configurations of the used JM17.2 High profile**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Entropy Coding</td>
<td>CABAC</td>
<td>ME Range</td>
<td>64</td>
<td>Profile/Level</td>
<td>High</td>
</tr>
<tr>
<td>8x8Transform</td>
<td>Enable</td>
<td>RDO</td>
<td>Used</td>
<td>Long-term</td>
<td>Enable</td>
</tr>
<tr>
<td>RDO Quant.</td>
<td>Enable</td>
<td>Ref Number</td>
<td>5</td>
<td>RDO Quant.</td>
<td>Used</td>
</tr>
<tr>
<td>SAD Method</td>
<td>hadamard</td>
<td>Intra Period</td>
<td>0</td>
<td>ME</td>
<td>UMH</td>
</tr>
<tr>
<td>Modes</td>
<td>All Used</td>
<td>Deblock</td>
<td>Enable</td>
<td>1/4-pel ME</td>
<td>Enable</td>
</tr>
</tbody>
</table>

**B. Background or Key Frame Updating**

Figure 13. Sequence structure for background generation.

Different kinds of background updating algorithms can be applied to our FET. Nevertheless, to highlight the transcoding efficiency, some factors such as the bit-allocation between background and input frames should not be taken into account in experiments, thus the background updating mechanism should be fixed and easy to implement. Therefore, the sequence structure in Fig. 13 is employed for background updating or key frame selection in all methods. In this structure, the background or key frame is updated periodically, and each background or key frame is transcoded by intra-prediction modes with the same quantization parameter. Moreover, each sequence is divided into super-groups of pictures (S-GOP’s). That is, an initial group of frames are utilized as TrainSet0 to update the background frame or select a key frame for S-GOP1, whereas the last group of frames in S-GOP1 are utilized as TrainSet1 for S-GOP2, and those in S-GOP2 are utilized as TrainSet2 for S-GOP3, ... Note that, the first frame is treated as the background or key frame for TrainSet0. In this way, each S-GOP owns the corresponding background frame for transcoding. In our experiments, the number of frames in each TrainSet is 120 and the length of an S-GOP is a function of the QP as

$$L = 300 \times \left[1 + \frac{1}{[Q P - 20] / 5}\right].$$

**VI. EXPERIMENTAL RESULTS**

Several experiments are designed to validate the efficiency of FET. Firstly, we present the distribution of FUs, FBUs and FBUs to show the effectiveness of transcoding with block classification in part A. Then, the performance gain, memory cost saving and time cost saving, which is brought by our proposed background modeling algorithm SWRA, are given in part B. Part C introduces the total bit saving and complexity saving results for FET-E/FET-EF over the stat-of-the-art methods. At last, a practical transcoding system is implemented based on open-source X264 video codec, the appearance and efficiency of the system can be seen from part D.

**Figure 14. Block category distribution for crossroad(CIF) and overbridge(CIF).**

**A. Block Classification Results**

**Table 7. The proportion of FUs, FBUs and FBUs in test sequences**

<table>
<thead>
<tr>
<th>Surveillance Videos: Proportion for each block category</th>
<th>SD bank office overbridge</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBU</td>
<td>10.07%</td>
<td>17.16%</td>
</tr>
<tr>
<td>FU</td>
<td>2.52%</td>
<td>5.11%</td>
</tr>
<tr>
<td>BU</td>
<td>87.41%</td>
<td>77.72%</td>
</tr>
</tbody>
</table>

**Table 7. The proportion of FUs, FBUs and FBUs in test sequences**

<table>
<thead>
<tr>
<th>Conferencing Videos: Proportion for each block category</th>
<th>CIF snowroad Snowgate overbridge</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBU</td>
<td>17.80%</td>
<td>16.47%</td>
</tr>
<tr>
<td>FU</td>
<td>0.95%</td>
<td>0.53%</td>
</tr>
<tr>
<td>BU</td>
<td>81.26%</td>
<td>83.00%</td>
</tr>
</tbody>
</table>

In the first experiment, we make a statistical analysis for the distribution of FUs, FBUs and FBUs. The result for each sequence can be seen from Table 7; the block category distributions for example frames of crossroad (CIF) and overbridge (CIF) are shown in Fig. 14. From these results, we can observe the BU’s take the largest part, so utilizing G-picture as long-term reference and designing specific speed-up techniques for BUs in FET will contribute a lot to the transcoding efficiency and complexity. Meanwhile, FBUs take much larger proportion than FUs. Thus after our FET saves the bit cost and complexity cost in BUs, thereby transcoding the large amount of FBUs will consume a large percentage of the total bit cost and time cost. In such case, our employed speed-up techniques for FBUs will save the complexity in further and G-picture will also reduce...
the bit-rates of FBUs by providing more accurate ground as reference. Moreover, although G-picture cannot provide more reference for FBUs, we can design speed-up strategies to reduce the candidate reference frames for FBUs to reduce the complexity. In summary, the statistics for block-category distributions indicates that designing category-adaptive speed-up strategies will be very effective for transcoding.

B. Experiment 2: Background Modeling Complexity and Efficiency of the Proposed SWRA

To evaluate the efficiency of the transcoding using SWRA, the FET-E using four state-of-the-art background modeling methods is employed as anchors for comparison, through their PSNR gains over T-AVC in high profile and IPPP structure on the first 620 frames of eight surveillance videos. The background modeling methods include the referred GMM-1, GMM-5, MS and RA. Transcoders using the methods are respectively namely FET-E-GMM-1, FET-E-GMM-5, FET-E-MS and FET-E-RA.

The transcoding performances of these anchors and our proposed FET-E-SWRA are firstly shown in Table 8, together with their background modeling time. It indicates that HP-GMM-X encoders seriously rely on the number of models utilized for each pixel. HP-GMM-1 achieves a much worse performance than other background modeling algorithms, and HP-GMM-5 achieves better performance than HP-RA, HP-MS and HP-GMM-1. On average, FET-E-SWRA achieves the best performance at 1.197/1.23 dB gain over HP on CIF/SD sequences. FET-E-SWRA is slightly better than FET-E-GMM-5, which achieves 1.17/1.22 dB gain. Besides, FET-E-MS is proved more efficient than FET-E-RA in LF sequences, but less efficient in SF ones. As for the modeling time, The RA escapes the least modeling time and MS escapes the largest. Moreover, it shows that SWRA escapes much less time than MS, GMM-1 and GMM-5 on all the sequences, only about 25% of the computing time spared by GMM-5. Moreover, SWRA is not sensitive to video content, which is quite different from GMM-X and MS.

Afterwards, another comparison for the memory cost of RA, MS, GMM-1 GMM-5 and proposed SWRA is shown in Table 9. Memory cost calculation for RA, MS, GMM-1 and GMM-5 has been referred in Sec. IV-A. In further for SWRA, the required memorized data for each pixel include: one current pixel with type of char and one float-precision mean value; two float-precision mean values avg/avg and their corresponding char-type weights. In summary, the total memory cost is no more than 14 bytes for each pixel. Results show that SWRA helps to achieve better performance than other modeling algorithm, on average. Moreover, compared to the state-of-the-art GMM-5, SWRA only consumes 10% of the memory cost and spares 25% of GMM-5’s modeling time.

C. Experiment 3: Total transcoding bit-rate and complexity

Table 10 lists the total PSNR gain and bit-rate saving of FET-E over T-AVC and T-KB for each sequence. This result can show the largest transcoding efficiency increase of the proposed FET. On average for surveillance videos, FET-E achieves bit-rate decreases of 39.84%/35.73% for SD/CIF sequences compared with T-AVC at the same PSNR, and 35.52%/27.50% over the state-of-the-art T-KB. These results correspond to 1.25/1.14dB gains over T-AVC, whereas 0.90/0.73dB PSNR gains over T-KB at the same bit-rate. For conference videos, FET-E achieves bit-rate decreases of 4.17%/5.85% for CIF/720p sequences compared with T-AVC at the same PSNR, and 3.34%/4.13% over the state-of-the-art T-KB. These results correspond to 0.20/0.20dB gains over T-AVC, whereas 0.16/0.14dB PSNR gains over T-KB at the same bit-rate. Firstly as we can see that, the less proportion of FBUs and BUs a sequence has, the less total bit-rate saving will be obtained (e.g. in surveillance videos, Crossroad (CIF) has the least proportion 66.53% and least bit-rate saving 17.06% over T-KB). This is because the performance gain of FET is mostly achieved on FBUs and BUs. Moreover, the transcoding efficiency increase of conference videos is much less than that of surveillance videos. This is because peoples in conference videos move slightly, so the exposed background regions will be much few than the surveillance videos in which cars or persons frequently cross the scene. Note that the transcoding efficiency RD curves of surveillance and conference videos are shown in Fig. 15 and 16. In summary, for surveillance and conference video transcoding, FET-E saves more than 35%/5% of the bit-rate of T-AVC. Compared to the T-KB, FET-E also saves about 30%/4% of the bit-rate, on average.

The comparison results of transcoding time for FET-EF vs. FET-E and FET-ES are shown in Table 11. These results show the largest complexity decrease of our FET over the FET-E without speed-up techniques and the FET-ES with state-of-the-art transcoding techniques. Because we have designed specific speed-up strategies for different block classifications in the motion compensation for FET-EF, the total time decrease is very large over the anchors. Before the comparison of transcoding time, we can discover from PSNR gains in Table 10 and 11 that, both the PSNR decreases of the FET-EF and FET-ES compared with FET-E are less than 0.1dB. This means the speed-up strategies still have similar PSNR gain with FET-E over T-AVC. Following this, as shown in Table 11, if we use Fast Full Search(FFS) for conference videos, FET-EF obtains as large as 15.4/7.5(CIF) and 22.3/12.0(720p) times speed up over FET-E/FET-ES, whereas the result is 16.1/7.9(CIF) and 10.0/5.3(SD) for surveillance videos. Otherwise, while Unsymmetrical-cross Multi-Hexagon grid Search(UMH) is used, for conference videos, the speed up is
coding the source video to four different lower bit-rate videos. The appearance of the systems can be seen from Fig. 17 and Fig.
18, where kinds of transcoding options, transcoding results and information are shown. For kinds of input high-definition sur-
veillance and conference videos in Fig. 19, the summarized performance of these systems can be shown in Table 13. Re-
results show that, this system can also averagely save 36.6% of the input four long-time H.264/AVC streams.

VII. CONCLUSION
In this paper, we propose a fast and efficient transcoding method (FET) for surveillance and conference videos based on
low-complexity background modeling and adaptive block classification. Results show that, FET averagely achieves more than
35% bit saving and larger than 10 times speed up over the traditional FDFE on the surveillance videos required high-efficiency transcoding. On the conference videos which should be transcoded to various devices with multiple band-
widths in real-time, FET can speed up more than 20 times and still achieve 0.2 dB transcoding performance gain over FDFE.
The main contributions of the proposed FET are illustrated as:

1) By theoretically analyzing what kind of background should be used and how the background should be quantized to
improve the efficiency, FET transcoded the modeled G-pictures into stream using specially designed QP and intra prediction.
And then, FET adopted the reconstructed G-pictures as long-term reference frames to significantly improve the transcoding efficiency of the following frames in surveillance and conference video. In our FET, G-picture was modeled from a
proposed low-complexity and high-efficiency background modeling algorithm.

2) Through analyzing the distributions of reference frames, motion vector and candidate prediction modes, FET proposed
to classify blocks into three categories by an adaptive block classification based on adaptively updating thresholds. And then, FET employed different speed-up strategies for different categories to dramatically save the transcoding complexity. These strategies were in forms of reference frame selection, ME search range reduction and candidate mode calculation.

3) Extensive experiments on kinds of surveillance and conference videos were utilized to evaluate the performance of
background modeling, block classification and the final transcoding efficiency and complexity. Moreover, FET was also
implemented in two practical systems respectively to transcode HD surveillance videos in lower bit-rates for dramatic storage
saving and real-time transcoding time. HD conference videos to various bit-rates for multiple-bandwidth transmission.

For future work, we will concentrate on accurate classification strategy and effective surveillance and conference video
analysis technology.

APPENDIX: PROOF FOR LEMMA 2
From the rate-distortion analysis for motion compensation in [24], the $\Phi(\Lambda)$ with two hypotheses (i.e., two reference frames) is related to the accuracy of motion compensation $P(\Lambda)$ by

$$\Phi(\Lambda) = \frac{(\Phi_{w_1}(\Lambda) + \Phi_{w_2}(\Lambda))}{4} + \frac{(\Phi_{w_1}(\Lambda) \times (3 + P(\Lambda)) + 2P(\Lambda) - 2P(\Lambda))}{2}. \quad (25)$$

In this equation, $\Phi_{w_1}(\Lambda)$ and $\Phi_{w_2}(\Lambda)$ are the $\Phi_w(\Lambda)$s for the two prediction hypotheses, whereas $P(\Lambda)$ and $P_2(\Lambda)$ are their corre-

D. Two practical systems based on FET
To practically assess the efficiency of our method, we also employ the proposed FET method to implement two practical
real-time transcoding systems for high-definition surveillance and conference videos. The first is a surveillance video trans-
coding system for decreasing the video bit-rate for storage. The second is a conference video transmission system for trans-

sponding $P(A)$. Let $\Phi(A)$ represent the $\Phi(A)$ using any long-term reference frame $i$, we can derive:

$$\Phi(A)=\Phi_{w,i}(A)+\Phi_{w,i}(A)/4+\Phi_{w,i}(A)(3+P_{i}(A)P_{i}(A)-2P_{i}(A)-P_{i}(A))/2,$$

where $P_{i}(A)$ and $P_{i}(A)$ denotes the $P(A)$ for the combination of short-term hypotheses and the long-term hypothesis, and $\Phi_{w,i}(A)$ and $\Phi_{w,i}(A)$ are corresponding $\Phi_{w,i}(A)$. Because any $\Phi(A)$ and $\Phi(A)$ use the same motion search, $P_{i}(A)=P_{i}(A)$, $\Phi_{w,i}(A)=\Phi_{w,i}(A)$ and $\Phi(A)=\Phi(A)$. Therefore, the difference $\Delta\Phi(A)$ between $\Phi(A)$ and $\Phi(A)$ is:

$$\Phi_{\Delta}(A)=\Phi_{w,i}(A)-\Phi_{w,i}(A)/4+\Phi_{w,i}(A)(3+P_{i}(A)P_{i}(A)-2P_{i}(A)-P_{i}(A))/2.$$

According to Girod et al. [25], $P_{i}(A)$ is determined by the displacement error variance $\sigma_{x}$ of the long-term reference frame $x$ and reflects the inaccuracy of the displacement vector used for the motion compensation. Therefore, when employing the same ME method, $P_{i}(A)=P_{i}(A)$.

From Eq. 27 and 28, we have

$$\Delta\Phi_{\Delta}(A)=\Phi_{w,i}(A)-\Phi_{w,i}(A)/4.$$

As pointed out by [14], $\Phi_{w,i}(A)$ is determined by the prediction error variance $\Phi_{\Delta}(A)$ in a monotone-increasing manner. For each block at position $(x, y)$ in $k$-th frame $I_k$, denote its PEV with $L_k$, as long-term reference and utilize a monotone increasing function $\Phi(I_k(x,y), L_k)$ to represent the $\Phi_{w,i}(A)$ with $I_k(x,y)$ as input. Then we re-write Eq. 29 as

$$\Delta\Phi_{\Delta}(A)=\Phi_{w,i}(A)-\Phi_{w,i}(A)/4.$$

In $OB$, noise and foreground pixels are much fewer than the $KB$ because of background generation. Besides, $OB$ also has much less quality loss than the $RB$ modeled from reconstructed frames. From the definition of $P(OB)$, we have

$$Z(I_k(x,y), OB, KB) = \begin{cases} \min \{Z(I_k(x,y), OB, KB), \} \\ \min \{Z(I_k(x,y), OB, KB), \} \end{cases}$$

From the monotone increasing property of $\Phi(I_k(x,y), L_k)$, we can further derive

$$\Phi(I_k(x,y), OB) \leq \min \{Z(I_k(x,y), OB, KB), \} \leq \Phi(I_k(x,y), OB, KB).$$

For a decoded surveillance or conference sequence, there are lots of background pixels in each $I_k$. Combining the cases in Eq. 32, we have

$$\sum_{i,j} \Psi(I_k(x,y), OB) \leq \Psi(\min \{Z(I_k(x,y), OB, KB), \} \leq \sum_{i,j} \Phi(I_k(x,y), KB)).$$

From Eq. 33 and 34, we can get

$$\Delta\Phi_{\Delta}(A) = \Phi_{w,i}(A)-\Phi_{w,i}(A)/4.$$

Therefore, we have $\Phi_{\Delta}(A)\leq\min\{\Phi_{w,i}(A), \Phi_{w,i}(A)\}$

REFERENCES


Table 10. FET-E vs. T-AVC/T-KB on overall bit-rate and PSNR (dB) on X86 platform (%)

<table>
<thead>
<tr>
<th>Surveillance videos</th>
<th>bank(SD)</th>
<th>office(SD)</th>
<th>overbridge(SD)</th>
<th>crossroad(SD)</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-AVC</td>
<td>FFS/UMH</td>
<td>PSNR</td>
<td>176.3/2.5</td>
<td>1.30</td>
<td>84/9.2/1.16</td>
</tr>
<tr>
<td>T-KB</td>
<td>FFS/UMH</td>
<td>PSNR</td>
<td>126.8/1.9</td>
<td>1.58</td>
<td>12.9/1.5</td>
</tr>
<tr>
<td>T-AVC</td>
<td>FFS/UMH</td>
<td>PSNR</td>
<td>12.3/1.5</td>
<td>-0.01</td>
<td>15.0/1.5</td>
</tr>
<tr>
<td>T-KB</td>
<td>FFS/UMH</td>
<td>PSNR</td>
<td>3.3/1.6</td>
<td>-0.02</td>
<td>3.0/1.7</td>
</tr>
<tr>
<td>T-AVC</td>
<td>Vidyo3</td>
<td>PSNR</td>
<td>52.6/2.4</td>
<td>0.20</td>
<td>23.9/1.7</td>
</tr>
<tr>
<td>T-KB</td>
<td>Vidyo4</td>
<td>PSNR</td>
<td>3.2/1.9</td>
<td>0.00</td>
<td>2.8/1.9</td>
</tr>
<tr>
<td>T-AVC</td>
<td>Vidyo1</td>
<td>PSNR</td>
<td>7.3/1.2</td>
<td>0.03</td>
<td>7.3/1.1</td>
</tr>
<tr>
<td>T-KB</td>
<td>Vidyo1</td>
<td>PSNR</td>
<td>3.2/1.8</td>
<td>0.00</td>
<td>3.2/1.7</td>
</tr>
<tr>
<td>T-AVC</td>
<td>Vidyo1</td>
<td>PSNR</td>
<td>65.6/2.1</td>
<td>0.14</td>
<td>65.6/2.0</td>
</tr>
<tr>
<td>T-KB</td>
<td>Vidyo1</td>
<td>PSNR</td>
<td>33.5/1.2</td>
<td>0.09</td>
<td>33.5/1.1</td>
</tr>
<tr>
<td>T-AVC</td>
<td>Vidyo1</td>
<td>PSNR</td>
<td>26.1/1.7</td>
<td>0.14</td>
<td>26.1/1.7</td>
</tr>
<tr>
<td>T-KB</td>
<td>Vidyo1</td>
<td>PSNR</td>
<td>43.4/1.7</td>
<td>0.16</td>
<td>43.4/1.7</td>
</tr>
</tbody>
</table>

Table 11. FET-E vs. FET-E/FES on Overall Transcoding speed up (times) and PSNR change (dB)

<table>
<thead>
<tr>
<th>Surveillance videos</th>
<th>carRoad(SD)</th>
<th>Crossroad(SD)</th>
<th>Office(SD)</th>
<th>GateGuard</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Search</td>
<td>95.3%</td>
<td>94.9%</td>
<td>99.5%</td>
<td>99.3%</td>
<td>99.3%</td>
</tr>
<tr>
<td>UMHexagon</td>
<td>65.8%</td>
<td>58.8%</td>
<td>71.9%</td>
<td>78.5%</td>
<td>78.5%</td>
</tr>
<tr>
<td>Conference</td>
<td>92.6%</td>
<td>97.8%</td>
<td>99.1%</td>
<td>98.7%</td>
<td>98.7%</td>
</tr>
<tr>
<td>Vidyo3</td>
<td>71.9%</td>
<td>78.3%</td>
<td>69.0%</td>
<td>81.6%</td>
<td>74.1%</td>
</tr>
<tr>
<td>Johnny</td>
<td>65.7%</td>
<td>81.8%</td>
<td>97.6%</td>
<td>99.7%</td>
<td>97.6%</td>
</tr>
<tr>
<td>Vidyo1</td>
<td>91.6%</td>
<td>99.6%</td>
<td>91.6%</td>
<td>99.5%</td>
<td>99.5%</td>
</tr>
</tbody>
</table>

Table 12. Search point reduction proportion using Full Search and UMHexagon algorithm

<table>
<thead>
<tr>
<th>Surveillance videos</th>
<th>FFS/UMH</th>
<th>PSNR</th>
<th>DU/SF/SM</th>
<th>FFS/UMH</th>
<th>PSNR</th>
<th>FFS/UMH</th>
<th>PSNR</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-AVC</td>
<td>160.0/1200</td>
<td>1600x1200</td>
<td>1920x1080</td>
<td>1920x1080</td>
<td>1920x1080</td>
<td>1920x1080</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 17. Example frames of videos for testing the transcoding system.

Figure 18. Surveillance video transcoding system for saving storage.

Figure 19. Conferencing video transcoding system for video transmission.
Xiaoguo Zhang (S’12) received the B.S. degree in computer science and technology from Peking University, Beijing, China, in 2007. From Sep. 2007, he is working toward the Ph.D. degree in computer application technology from the Department of Computer Science and technology. Currently, he is a student in the National Engineering Laboratory for Video Technology, School of Electronics Engineering and Computer Science, Peking University. His research interests include video coding, transcoding and processing.

Tiejun Huang (M’01–SM’12) is a professor of the School of Electronic Engineering and Computer Science, Peking University, and the vice director of the National Engineering Laboratory for Video technology of China. Prof. Huang received Ph.D. degree on Pattern Recognition and Image Analysis from Huazhong (Central China) University of Science and Technology in 1998 and master and bachelor degree on computer science from Wuhan University of Technology in 1995 and 1992. His research area includes video coding, image understanding, digital right management (DRM) and digital library. He published more than sixty peer-reviewed papers and three books as author or co-author. He is the member of the Board of Director for Digital Media Project, Advisory Board of IEEE Computing Now, Editorial Board of Springer Journal on 3D Research and the Board of Chinese Institute of Electronics.

Yonghong Tian (M’05–SM’10) received the Ph.D. degree from Institute of Computing Technology, Chinese Academy of Sciences, China, in 2005. He is currently a Professor with the National Engineering Laboratory for Video Technology, School of Electronics Engineering and Computer Science, Peking University, Beijing, China. He is the author or co-author of over 80 technical articles in refereed journals and conferences. His research interests include computer vision, multimedia analysis and coding. Dr. Tian is currently a Young Associate Editor of the FRONTIERS OF COMPUTER SCIENCE IN CHINA. He was the recipient of the Second Prize of National Science and Technology Progress Awards in 2010; the best performer in the TRECVID content-based copy detection (CCD) task (2010-2011); the top performer in the TRECVID retrospective surveillance event detection (SED) task (2009-2012); the winner of the WikipediaMM task in ImageCLEF 2008.

Mingchao Geng received his B.S. degree in electronic technology from Shandong University, Weihai, China, in 2009, and the M.S. degree in electronic science and technology from Peking University, Beijing, China, in 2012. When worked as a student in the National Engineering Laboratory for Video Technology, he focused on fast surveillance video transcoding methods and practical transcoding systems.

Siwei Ma (S’03–M’12) received the B.S. degree from Shandong Normal University, Jinan, China, in 1999, and the Ph.D. degree in computer science from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2005. From 2005 to 2007, he was a Post-Doctorate with the University of Southern California. Then he joined the Institute of Digital Media, EECS, Peking University, where he is currently an Associate Professor. He published over 100 technical articles in refereed journals and proceedings in the areas of image and video coding, video processing, video streaming, and transmission.

Wen Gao (M’92–SM’05–F’09) received the Ph.D. degree in electronics engineering from the University of Tokyo, Tokyo, Japan, in 1991. Currently, he is a Professor of computer science at Peking University, Beijing, China. Before joining Peking University, he was a Professor of computer science with the Harbin Institute of Technology, Harbin, China, from 1991 to 1995, and a Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing.