A CODING UNIT CLASSIFICATION BASED AVC-TO-HEVC TRANSCODING WITH BACKGROUND MODELING FOR SURVEILLANCE VIDEOS

Peiyin Xing[a,b], Yonghong Tian[b*], Xianguo Zhang[b], Yaowei Wang[c], Tiejun Huang[b]
[a]The Shenzhen Key Lab for Cloud Comput. Tech. & App., Shenzhen Graduate School, Peking University, Shenzhen 518055, P.R. China
[b]Institute of Digital Media, Peking University, Beijing, 100871, P.R. China
[c] Department of Electronic Engineering, Beijing Institute of Technology, Beijing 100081, China
[*] Corresponding to: yhtian@pku.edu.cn

ABSTRACT

To save the storage and transmission cost, it is applicable now to develop fast and efficient methods to transcode the perennial surveillance videos to HEVC ones, since HEVC has doubled the compression ratio. Considering the long-time static background characteristic of surveillance videos, this paper presents a coding unit (CU) classification based AVC-to-HEVC transcoding method with background modeling. In our method, the background frame modeled from originally decoded frames is firstly transcoded into HEVC stream as long-term reference to enhance the prediction efficiency. Afterwards, a CU classification algorithm which employs decoded motion vectors and the modeled background frame as input is proposed to divide the decoded data into background, foreground and hybrid CUs. Following this, different transcoding strategies of CU partition termination, prediction unit candidate selection and motion estimation simplification are adopted for different CU categories to reduce the complexity. Experimental results show our method can achieve 45% bit saving and 50% complexity reduction against traditional AVC-to-HEVC transcoding.

Index Terms— video transcoding, surveillance video, HEVC, coding unit classification, background modeling

1. INTRODUCTION

In surveillance applications, large storage and bandwidth cost is required to record and transmit the long-period video archives. To save the cost, it is a reasonable solution to transcode surveillance videos using a high-efficient encoding process. Recently, the latest video coding standard, High Efficiency Video Coding (HEVC) [1], can achieve about 50% bit-rate reduction against its predecessor H.264/AVC [2] (shorten for AVC) at the same perceptual quality. Therefore, it is very meaningful to transcode surveillance videos from AVC to HEVC. However, the efficient quadtree based CU partition and various patterns of prediction unit (PU) in HEVC also remarkably increase the encoding complexity. Consequently, it is desired to develop higher-efficiency and lower-complexity technologies to transcode the widely used AVC surveillance video streams to HEVC ones.

Among the transcoding methods, directly connecting transcoder from the source-format decoder and target-format encoder can be named full decoding and full encoding (FDFE). Although FDFE is considered to be the most efficient, it is not practical due to the high computational complexity. Therefore, various speed-up transcoding techniques have been investigated in [3][4][5]. For example, Shin et al. [3] developed a motion vector (MV) clustering method to accelerate the motion estimation (ME) procedure.

While transcoding from AVC to HEVC, the different partitions of coding units between AVC and HEVC make the reuse of motion vector and coding modes much more complex. To address the problem, a motion vector reuse method was introduced by Peixoto et al. [6], in which CU size and PU pattern are determined by the similarity among the corresponding decoded MVs in AVC stream. Moreover, D. Zhang et al. [7] proposed a fast CU partitioning and PU candidate selection transcoding procedure. They estimated the best CU split quadtree, PU mode and MV of each PU by utilizing the power spectrum based rate-distortion (RD) optimization model. However, none of the referred methods [3-7] is specially designed for surveillance videos. Intuitively, if typical characteristics of surveillance videos (e.g., the long-time static background) can be exploited, better transcoding efficiency can be achieved.

Thus to obtain a high-efficiency and low-complexity surveillance video transcoding from AVC to HEVC, we propose a coding unit classification based AVC-to-HEVC transcoding method with background modeling (namely CTBM) for surveillance videos in this paper. For efficiency, we propose to embed background modeling into AVC-to-HEVC transcoder, where the beginning originally decoded frames are utilized to model a background frame. Afterwards, the modeled background frame is transcoded into HEVC stream as long-term reference to enhance background prediction efficiency of the following frames.

As for complexity, a CU classification algorithm is firstly developed using the decoded motion vectors and the modeled background frame as input. As a result, the decoded data are classified into background CUs (BCs, mainly background pixels), foreground CUs (FCs, mainly foreground pixels) and hybrid foreground and background CUs (HCs). The statistics on each CU category shows that different kinds of to-be-transcoded CUs tend to own different characteristics in the HEVC recursive coding structure, such as different terminated depths of CU partitioning, different PU patterns for inter prediction etc. Inspired by this, we propose to adopt different transcoding strategies to transcode different CU categories of decoded data into HEVC streams (the streams can still be decoded by HEVC decoder). These strategies include CU Partition Determination, PU Candidate Selection, and ME Simplification.

Experimental results show that our method can achieve 49.9% (CIF) and 54.8% (SD) transcoding time reduction against traditional FDFE (T-FDFE, directly combining AVC decoder and HEVC encoder),with 44.6% (CIF) and 46.5% (SD) bit saving.

The rest of this paper is organized as follows: Section 2 analyzes the problems of transcoding from AVC to HEVC. Section 3 gives an overview of our transcoding method. Section 4 shows the experimental results. Section 5 concludes the paper.

2. PROBLEM ANALYSIS

HEVC still follows the traditional "hybrid" encoding method (inter-/intra-picture prediction and 2D transform coding) used in all previous compression standards. Nevertheless, it also introduces some new coding tools. Among them, the quadtree based block partition is one of the most important changes with dramatic impact on efficiency and complexity.

Referred to the partition, novel concepts are introduced, including Coding Unit (CU), Prediction Unit (PU) and Transform Unit (TU). CU is the basic processing unit rather than Macroblock (MB) in previous standards such as AVC. The size of PU and TU depends on the size of CU. Instead of dividing each picture into MBs, HEVC partitions each picture into CUs, which are squared regions with size of 2N×2N. The largest CU size is 64×64(N=32) and the smallest is 8×8(N=4). Fig. 1 shows the recursive quadtree CU partitioning process. For each CU, the candidate patterns of PU for inter prediction include symmetric partitions of 2N×2N, 2N×N, N×2N, N×N and asymmetric motion partitions (AMP) of 2N×nU, 2N×nD, nL×2N and nR×2N, all of which are illustrated in Fig. 2. Note that, the best CU partition and PU candidate will be selected through the mode decision process. It is obvious that this decision process will be very costly. Beside above features, another factor impacting the real-time surveillance video transcoding is the low-delay reference frame selection. Different from AVC's reference software Joint Model (JM), HEVC's reference Model (HM) selects the previous frame and the last frames of previous Group of Pictures (GOPs) as reference frames. Taking four reference frames and GOP size equal to 4 as an example, the picture distances between the current frame and its reference frames in JM's low-delay encoder are -1,-2,-3,-4, and those for HM's low-delay encoder are -1,-5,-9,-13.

In summary, the different size of coding units, prediction units and reference frame structure make it difficult to directly reuse the decoded motion vectors, prediction modes and reference frames from AVC decoder. Thereby the transcoding method from AVC to HEVC should be more complicated to realize remarkable complexity reduction. For surveillance video transcoding, the specific characteristics of long-time static background can be exploited for time saving. A reasonable idea follows: after classifying the decoded data using the modeled background frame, CU-category adaptive fast strategies may be summarized to reduce the transcoding complexity.

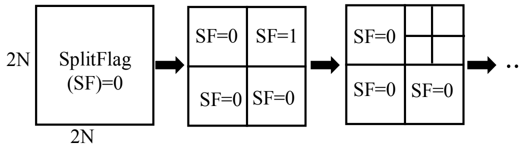


Fig. 1. CU partitioning

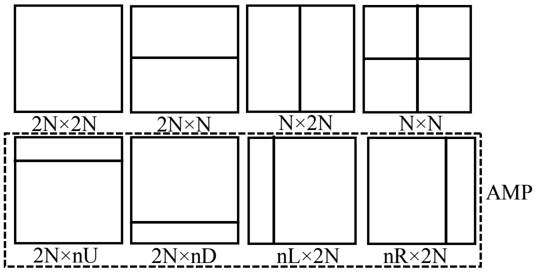


Fig. 2. PU patterns of inter prediction

2.1. Experimental setup for problem analysis

To investigate the detailed fast transcoding strategies, experiments are conducted in this section to analyze the distributions of CU partition, PU candidate patterns, motion vectors and reference frames. The experimental platform is the HM8.0 using the modeled background as long-term reference. Because surveillance videos are always recorded real-timely, the low delay main configuration [8] is utilized to configure HM8.0. While transcoding, the source videos are crossroad-cif, bank-cif, campus-cif and classover-cif (referred in Sec. 4) compressed by JM17.2 baseline profile with quantization parameter (QP) equal to 16. During the transcoding result analysis, the transcoding information is made respectively for BCs, FCs and HCs on the four CIF sequences. Note that, the background modeling and CU classification algorithm will be referred in Sec. 3.

2.2. Analysis of CU partition

Table 1 shows the distribution of CU partition for different CU categories. As is seen, the "split CUs" take up 8.46% on average for BC, and the proportion is much less than that of FC and HC. However, the proportion 8.46% is not very small and directly terminating CU partition for BCs may lead noticeable performance loss. Therefore, we make the following further analysis. Denoting the CUs without any foreground Basic Unit (foreground BU, which is a 4x4 block with seldom background pixels and defined in Sec. 3) as pure background CUs, we can figure out its spilt and non-spilt proportion in Fig. 3. It can be observed that 98.30% of pure background CUs will not split any more. In other words, only 1.70% of pure background CUs with depth equal to t will be split to depth t+1. Therefore, we can summarize the *CU partition determination rule*: CU partitioning can be early terminated if current CU is a pure background.

Table 1. The distribution of CU partition of BC, HC and FC

CU category	BC	HC	FC
non-split(SF=0)	91.54%	43.45%	39.96%
split(SF=1)	8.46%	56.55%	60.04%

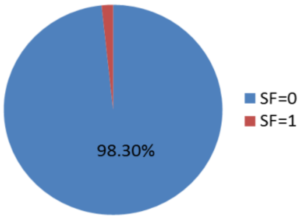


Fig. 3. The proportion of split and non-split pure-background CUs

2.3. Analysis of PU patterns

It is apparent that the proportion of each prediction mode varies among BC, HC and FC. Table 2 shows the distribution of the prediction modes of BC, HC and FC. As can be seen from the table, 97.30% of BCs select 2N×2N inter prediction mode. The selection of 2N×N, N×2N and N×N account 1.52%, and that for AMP account 1.18%. Moreover, from BC, HC to FC, the percentage for 2N×2N inter prediction mode decreases, and those for 2N×N&N×2N&N×N and AMP patterns enlarge. Consequently, we can get *PU candidate selection rule*: only 2N×2N mode should be used for BCs, the AMP prediction modes are disabled for HCs and all the candidate modes will be tried for FCs.

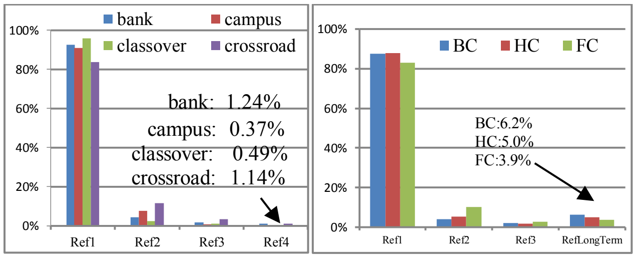
Table 2. The distribution of prediction modes of BC, HC and FC

CU category	2N×2N	2N×N&N×2N&N×N	AMP
BC	97.30%	1.52%	1.18%
HC	96.51%	2.40%	1.09%
FC	93.64%	3.72%	2.64%

2.4. Analysis of motion estimation

For BC, HC and FC, the used reference frames probably have a different distribution. In order to find the best way to select reference frames, we analyze the distribution of the selected reference frames with and without long-term reference. For BC, HC and FC, Fig. 4 (a) and (b) respectively depicts the distributions without long-term reference (HM8.0 without background modeling) and that with the modeled background frame as long-term reference. As can be seen, both Fig. 4 (a) and (b) show the importance of the first reference. Moreover in Fig. 4 (a), the fourth reference frame only accounts a small percentage of 0.81% on average. Whereas in Fig. 4 (b), the fourth background reference takes up larger than 5% for BC and HC, the third reference for all categories and the second reference for BC is less than 5%. In summary, the first and long-term reference are required for BC; the first, second and long-term reference are needed for HC; while the first and second are necessary for FC.

Besides reference frame selection, motion search range is another important factor in ME that affects the complexity. Intuitively, the transcoding search range should be no less than the so-called best MVD, the difference between predicted motion vector (PMV) and the best matched motion vector. Table 3 shows the distribution of best MVDs of BC, HC and FC. From the table, we can find that more than 99% of the best MVDs are no more than 1 pixel for BC even in the crossroad-cif which has lots of moving blocks.



(a) (b)

Fig. 4. The distribution of reference frames of T-FDFE:
(a) without long-term reference; (b) with long-term reference

Table 3. The distribution of MVDs

mvd range CU category	<=1pixel	1pixel~4pixel	>4pixel
BC	99.671%	0.328%	0.001%
HC	99.089%	0.909%	0.002%
FC	98.326%	1.656%	0.018%

Moreover, five candidates will be checked in the process of start-search-position selection for ME in HM, which enlarge the ME time. These candidates are: the motion vector predictor (PMV) obtained by motion vector predictor derivation process, three motion vectors of neighboring positions and zero motion [9]. Actually, some candidates for some CU categories can be skipped using the decoded MVs.

From the analyses above, we can summarize the *motion estimation simplification rule:* For BCs, the second and third reference frames can be forbidden; the first, second and long-term reference frames are used for HCs; for FCs, only the first and second reference frames will be used. The motion search range will be just set to 1 pixel for BCs, and for HCs and FCs, motion search range should be larger in order to maintain the performance. Besides, some start search positions should be skipped.

3. THE PROPOSED METHOD

Following the summarized rules for transcoding AVC surveillance streams to HEVC ones, we propose a coding unit classification based AVC-to-HEVC transcoding method with background modeling for surveillance videos in this section. As depicted in Fig. 5, CTBM is composed of Background Modeling and Encoding, CU Classification, CU Partition Determination, PU Candidate Selection and ME Simplification. By adopting these, our CTBM works as follows:

1) A background frame is modeled from originally decoded frames and then encoded as long-term reference.

2) Each CU is classified into BC, HC or FC by calculating the differences between itself and the corresponding background data using threshold judgment.
3) With the help of the CU classification information and the decoded prediction mode information, we early-terminate CU partition or decide the PU size in advance.
4) BC, HC and FC are processed by the ME Simplification module respectively. The process includes reference frame selection, search range modification and start-search-position refinement using the decoded reference frames and motion vectors from AVC decoder.

Details about the modules will be referred in the following.

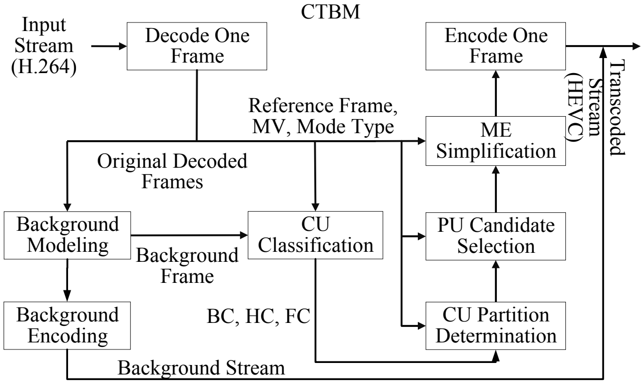


Fig. 5. The framework of the proposed method

3.1. Background modeling, updating and encoding

Considering the transcoding time and memory cost, the low complexity segment-and-weight based running average in X. Zhang et al [10] is utilized as the background modeling algorithm. In general, this method models a background value of pixels at each position by the following five steps: initializing average values and corresponding weights, calculating the threshold for temporal segmenting, creating a new segment or widen the current segment, updating the average values or calculating the final background value. While transcoding, the background frame should be updated to provide a better long-term reference. To avoid the bit-allocation problem, we still follow [10] to update each background frame every super group of L frames, the last M frames of which are used to generate the background frame to predict the following super group of pictures. In this way, a no-delay coding can be guaranteed. In order to produce a high-quality background as long-term reference and guarantee the decoding match, the modeled background frame is encoded into HEVC stream and only intra prediction and the decoded minimum QP are employed.

3.2. CU Classification

After the background frame is generated, we classify a CU according to the foreground or background properties of its inside BUs. To judge the property of a BU, a reasonable idea is to calculate the difference between the BU and its corresponding BU in the background frame. Besides, if one decoded BU has a large motion vector, it should also be classified as foreground BU.

Let us donate $b_{i,j}$ as the pixel value at row i and column j in current BU and $BG_{i,j}$ as the pixel value at the corresponding position in the modeled background frame. We use (mv_x, mv_y) as the MV obtained from AVC decoder. Then we can calculate the property $P(b)$ of a BU b as background B or foreground F by

$$P(b)=\begin{cases} F, & \sum_{i=1}^{4}\sum_{j=1}^{4} abs(b_{i,j} - BG_{i,j}) > \alpha \ or \ \sqrt{mv_x^2+mv_y^2} > v \\ B, & \sum_{i=1}^{4}\sum_{j=1}^{4} abs(b_{i,j} - BG_{i,j}) \le \alpha \ and \ \sqrt{mv_x^2+mv_y^2} \le v \end{cases}. \quad (1)$$

This means current BU will be judged as foreground if sum of background difference exceeds the threshold value α (80 in our experiment) or it has obvious motion (v=2 in our experiment). Otherwise, it will be judged as background. With the properties of inside BUs, each CU can be classified according to the distributions of inside BUs.

A most direct idea for CU classification is to judge whether the proportion of foreground or background BUs exceed a threshold. However, even if the above condition is satisfied, there are still exceptions. For example, one CU has relative large proportions of background BUs, but there are still some foreground BUs clustering together. In such case, the CU cannot be classified into BC, since a worse coding of foreground BU quality will significantly decrease the total coding performance. Therefore, we should add some constrains beyond counting the proportion of BUs, e.g., with the help of decoding information and distributions of BUs. Following this idea, we firstly denote any two neighboring decoded foreground BUs ($P(b)$=F) as a foreground group (FG). Moreover, if one foreground BU is neighboring to any BU in a foreground group, it will be added to the FG. This procedure will iterate until size of the FG never enlarges.

With each BU's property and all foreground groups' information, we can obtain each CU's category C(c) through calculating and comparing the proportion of foreground BUs of current CU c. Supposing $\|X\|$ represents the size of a set X, $b(i)$ is the i-th BU in c, $fg(j)$ is the j-th FG in c, and 2N×2N is the size of c, the calculation process is :

$$C(c)=\begin{cases} FC, & 16\times\left\|\{i \mid P(b(i))=F\}\right\|/N^2 > \delta \ and \ \exists j \ st. \ 32\times\left\|fg(j)\right\|/N^2 > \delta \\ BC, & 16\times\left\|\{i \mid P(b(i))=F\}\right\|/N^2 \le \varepsilon \ and \ \forall j \rightarrow 32\times\left\|fg(j)\right\|/N^2 \le \varepsilon \\ HC, & others \end{cases}. \quad (2)$$

where δ is practically set to 0.5 and ε is 0.0625. All the thresholds are obtained from the analysis experiment. The classification of BC, HC and FC will be more consistent with the scene content using these thresholds. Following these, if foreground-BU proportion is no more than 1/16 and no FG takes up more than 1/32 of the total BU number, the current CU will be categorized as BC; If the foreground-BU proportion is more than 1/2 and there is one FG, covering more than 1/4 BUs, the CU is classified to FC; Otherwise it should be an HC. Algorithm 1 describes the above CU classification procedure.

```
Algorithm 1: CU classification algorithm
Input value: Current CU c,
Output: current category class C(c) in {FC,HC,BC}
Classification procedure:
A. Calculate the proportion of foreground BUs R in c:
  $R=16\times\left\|\{i\mid P(b(i))=F\}\right\|/N^2$
B. Find every foreground group
  $i$=1,$j$=1,$k$=1, $\exists m, \mathrm{P}(b(m))=F$, Then $fg(1)=\{b(\mathrm{m})\}$
  While (1)
    For $k$=1 to $k=j$
      For $i$=1 to $i=N^2/4$
        If $\exists x\exists k, b(i)$ neighbors $b(x), P(b(i))=F \wedge b(i)\notin fg(k)\wedge b(x)\in fg(k)$
        Then $fg(k)=\{b(i)\}\cup fg(k)$
    If $\exists x, P(b(x))=F\wedge b(x)\notin fg(1)\cup\ldots\cup fg(j)$ Then $j$++, $fg(j)=\{b(x)\}$
    Else Break
C. Classification
  If $R>\delta$ and $\exists j\ st.\ \left\|fg(j)\right\|/N^2>\delta/32$, Then C($c$) =FC
  Else if $R\le\varepsilon$ and $\forall j\rightarrow\left\|fg(j)\right\|/N^2\le\varepsilon/32$, Then C($c$) =BC
  Else C($c$) =HC
```

3.3. CU Partition Determination

As discussed in Sec. 2.2, most of the pure background CUs will not split any more. Thus following the *CU partition determination rule*, if the current CU is a pure background CU, the recursive CU partitioning will be terminated. Furthermore, if CU size is 16×16 (equal to that of a MB), the decoded mode of SKIP or P16×16 will indicate the pixels of the decoded block have similar motion. In such case, it is reasonable to terminate the CU partition. Fig. 6 depicts the CU Partition Determination process.

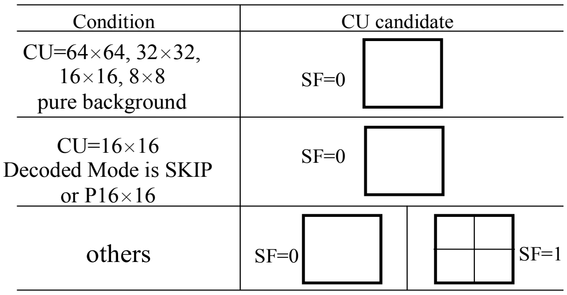

Condition
CU candidate
CU=64×64, 32×32, 16×16, 8×8 pure background
SF=0
CU=16×16 Decoded Mode is SKIP or P16×16
SF=0
others
SF=0
SF=1

Fig. 6. CU Partition Determination process

Table 4. Prediction mode candidates for different CU categories

category	Prediction mode candidate
BC	Merge,2N×2N
HC	Merge,2N×2N,2N×N,N×2N,N×N
FC	All

3.4. PU Candidate Selection

As summarized in Sec. 2.3, the large size prediction modes will be selected mostly for static region. Therefore, following the *PU candidate selection rule*, we only chose 2N×2N for BCs and other PU sizes will be skipped. On the contrary, all the possible prediction modes will be tried for FCs. And for HCs, only AMP prediction modes are disabled. Prediction mode candidates for every category are listed in Table 4.

3.5. ME Simplification

The long-term reference frame takes up a higher proportion in BCs than HCs and FCs. To refine the reference frame selection, we skip some reference frames according to the correlation and analysis of these three categories. Beyond the *motion estimation simplification rule*, the reference frames decoded from the bit stream are also added to the reference frame candidate pool to maintain the performance. In summary, the reference frame set *S*(*c*) for the current CU *c* is described in Eq. 3.

$$S(c)=\begin{cases}\{R0,Bg\}U\{\lfloor i/GopSize\rfloor\},C(c)=BC\\ \{R0,R1,Bg\}U\{\lfloor i/GopSize\rfloor\},C(c)=HC,\\ \{R0,R1\}U\{\lfloor i/GopSize\rfloor\},C(c)=FC\end{cases} \tag{3}$$

where *R0*, *R1*, *R*2 and *Bg* represent the first, second, third and long-term reference frame respectively, *i* is the reference frame index obtained from AVC decoder and *GopSize* is the length of GOP in HEVC encoder.

To further reduce complexity, as said in the *motion estimation simplification rule*, motion search range will be set to 1 pixel for BCs, and the range will be modified to the maximum MVD obtained from AVC decoder for HCs and FCs. As for the start-search-position selection, we propose to skip test zero position if all the MVs getting from AVC decoder is not zero. Equation 4 shows the algorithm for skipping test zero MV start search position. In Eq. 4, *SP* is the search start position set; *A*, *B* and *C* are the three motion vectors of neighboring positions; *Zero* is the zero position; *MV* is the decoded motion vectors' set of the current CU.

$$SP=\begin{cases}\{PMV,A,B,C,Zero\},\exists mv\in MV => mv=\mathbf{0}\\ \{PMV,A,B,C\},\ Otherwise\end{cases}. \tag{4}$$

bank-cif campus-cif classover-cif crossroad-cif
bank-sd campus-sd classover-sd crossroad-sd

Fig. 7. Examples of surveillance videos used to evaluate CTBM

4. EXPERMENTAL RESULTS

To verify the performance of our transcoding method, we compare the efficiency and complexity of our method with T-FDFE. Both T-FDFE and our CTBM are implemented on HM8.0 under the low-delay-main common test conditions [8] for the real-time surveillance videos. Moreover, the experimental dataset, eight CIF&SD surveillance videos with 3000 frames, are compressed by AVC software JM17.2 with baseline configuration and QP=16, most of which have been utilized to evaluate the method in [10]. They will be transcoded by CTBM and T-FDFE with QP=22, 27, 32, 37. Fig.

7 shows the examples of the CIF and SD surveillance videos with different motion characteristics.

4.1. Efficiency and complexity analysis

For efficiency, experimental results in Table 5 show that CTBM can save 49.9% (CIF) and 54.8% (SD). In detail, we can find that larger bit-rate saving will be achieved on the sequences with larger background regions. For example, the crossroad-cif that has lots of moving cars has the least bit-saving and the static bank-cif gains the most bit-saving. This result reveals the significant improvement in transcoding efficiency is mainly produced by the background modeling.

For complexity reduction, Table 5 also shows transcoding time is reduced by 44.6% (CIF) and 46.5% (SD) over T-FDFE. Also in detail, the more background regions a video has, the more complexity reduction is achieved. This is because our CU category adaptive transcoding strategy makes more efforts on the time saving for BCs, in which faster CU partition termination, PU candidate selection and ME simplification are designed.

It should be noted that, any fast algorithm will produce some quality loss than the method without the fast algorithm. However as Table 6 shows, the fast transcoding algorithms of CU Partition Determination, PU Candidate Selection and ME Simplification in CTBM only produces 3.3% (CIF)/3.7% (SD) loss on average compared with the CTBM only using background modeling based efficiency optimization. Fig. 8 depicts the transcoding RD curves and time saving example.

Table 5. Performance and complexity comparison between CTBM and T-FDFE on CIF and SD sequences

Sequence	BD rate	PSNR gain	Time	BD rate	PSNR gain	Time
	CIF			SD		
bank	-60.2%	1.030 dB	-37.5%	-67.6%	1.644 dB	-65.6%
campus	-52.0%	1.241 dB	-48.6%	-49.1%	1.168 dB	-48.6%
classover	-42.4%	1.045 dB	-61.5%	-45.0%	1.209 dB	-67.4%
crossroad	-23.6%	0.840 dB	-51.8%	-24.3%	0.811 dB	-37.6%
average	-44.6%	1.039 dB	-49.9%	-46.5%	1.208 dB	-54.8%

Table 6. PSNR loss produced by the fast transcoding algorithm

Sequence	BD rate	PSNR gain	BD rate	PSNR gain
	CIF		SD	
bank	2.0%	-0.048 dB	3.3%	-0.115dB
campus	2.4%	-0.088 dB	3.0%	-0.090 dB
classover	4.2%	-0.141 dB	4.0%	-0.133 dB
crossroad	4.8%	-0.190 dB	4.5%	-0.166 dB
average	3.4%	-0.117 dB	3.7%	-0.126 dB

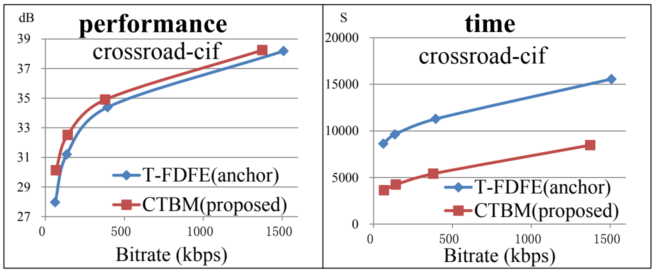


Fig. 8. RD curves and time saving example

4.2. Additional experiments

Additionally, we also analyze the complexity reductions separately performing each fast transcoding strategies in our CTBM. As Table 7 shows, the complexity reductions are respectively 10.57%, 20.19% and 26.48% for CU Partition Determination, PU Candidate Selection and ME Simplification. This reveals the importance of each strategy.

Table 7. The complexity reduction while separately performing each fast transcoding strategy

Strategy	CU Partition Determination	PU Candidate Selection	ME Simplification
Time saving	10.57%	20.19%	26.48%

5. CONCLUSION

In this paper, we proposed a coding unit classification based fast and efficient AVC-to-HEVC transcoding method for surveillance videos. Beside the more efficient background prediction from background modeling, a CU classification algorithm using the modeled background is proposed to transcode the decoded data into HEVC streams of CU categories (BC, FC and HC) with different fast transcoding strategies. Experimental results showed that CTBM could averagely reduce the total transcoding time by 49.9% (CIF) and 54.8% (SD) on the eight surveillance sequences, with 44.6% (CIF) and 46.5% (SD) bit saving over the traditional FDFE without background modeling. For the future work, we will focus on more accurate CU classification and more sufficient utilization of decoding information.

ACKNOWLEDGEMENT

This work is partially supported by grants from the Chinese National Natural Science Foundation under contract No. 61035001, 61121002 and 61176139.

6. REFERENCES

[1] G. J. Sullivan, W. Han, “Overview of the High Efficiency Video Coding (HEVC) standard,” in T-CSVT, Dec. 2012.

[2] T. Wiegand, G. J. Sullivan and et al., “Overview of the H.264 video coding standard,” in T-CSVT, Jul. 2003.

[3] Y. Shin and et al., “Low-complexity heterogeneous video transcoding by motion vector clustering,” in ICISA, April 2010.

[4] A. Vetro, C. Christopoulos, and H. Sun, “Video transcoding architectures and techniques: an overview,” IEEE Signal Process. Mag., vol. 20 (2), pp. 18-29, Mar. 2003.

[5] I. Ahmad and et al., “Video transcoding: an overview of various techniques and research issues,” in T-MM, Oct. 2005.

[6] Eduardo Peixoto and et al., “A complexity-scalable transcoder from H.264 to the new HEVC codec,” in ICIP, pp. 737-740, Sept. 2012.

[7] D. Zhang, B. Li, J. Xu and et al. “Fast transcoding from H.264 to High Efficiency Video Coding,” in ICME, July 2012.

[8] “HM 8 common test conditions and software reference configurations,” in JCTVC-J1100, Jul. 2012.

[9] “High Efficiency Video Coding (HEVC) Test Model8 (HM8) encoder description,” in JCTVC-J1002, Jul. 2012.

[10] X. Zhang and et al., “Low-complexity and high-efficiency background modeling for surveillance video coding,” in VCIP, Nov., 2012