A NOVEL SCREEN CONTENT FAST TRANSCODING FRAMEWORK BASED ON STATISTICAL STUDY AND MACHINE LEARNING

Fanyi Duanmu[1], Zhan Ma[2], Wei Wang[3], Meng Xu[4] and Yao Wang[1]

[1]New York University, Tandon School of Engineering, Brooklyn, NY 11201, USA;
[2]Nanjing University, 22 Hankou Road, Nanjing 210093, China;
[3] FutureWei Technologies Inc., 2330 Central Expy, Santa Clara, CA 95050, USA;
[4]Real Communications Inc., 2870 Zanker Road #110, San Jose, CA 95134, USA.
(fanyi.duanmu@nyu.edu, mazhan@nju.edu.cn, wei.wang.sc@huawei.com, meng.xu@realcomtec.com, yw523@nyu.edu)

ABSTRACT

In this paper, a novel screen content transcoding framework is presented to efficiently bridge the state-of-art High Efficiency Video Coding (HEVC) standard and its incoming screen content coding (SCC) extension currently pending finalization. The proposed scheme is implemented as an Intra-coding "pre-processing" module on top of official SCC test model software (SCM). Both Coding Unit (CU) statistical features (such as CU color quantity, CU pixel variance, CU edge directionality distribution, etc.) and decoded video side information (such as CU partitions, modes, residual, etc.) are jointly analyzed. Accordingly, fast CU mode decisions and CU partitions bypass / termination heuristics are designed. Compared with SCM-4.0 official release, the proposed fast transcoding scheme can achieve an average of 48% re-encoding complexity reduction over JCT-VC screen content testing sequences with less than 2.14% marginal BD-Rate increase under SCC common testing conditions for All-Intra (AI) configuration.

Index Terms - High Efficiency Video Coding (HEVC), Screen Content Coding (SCC), Video Transcoding (VTC), Fast Mode Decision (FMD), Machine Learning (ML).

1. INTRODUCTION

Screen content (SC) videos have become very popular due to the recent advances in mobile and cloud applications, such as online education, remote desktop collaborations, virtual desktop interfacing, cloud gaming, wireless display, etc. Such emerging applications and market demands create an urgent need for more efficient compression and delivery of screen content videos. To address these topics, the Joint Collaborative Team on Video Coding (JCT-VC) launched the standardization of SCC extension [1] since January 2014 on top of the latest HEVC standard, to put together SCC research efforts from both academia and industry.

So far, the standardization is pending finalization by early 2016. The latest SCC Test Model software (i.e., SCM), is reported to provide over 45% BD-Rate saving beyond HEVC Range Extension (RExt) [1] for typical computer-generated contents. This significant gain is attained from 4 novel coding tools beyond HEVC, known as "Intra Block Copy" (IBC) [2] [3], "Palette Coding Mode" (PLT) [4], "Adaptive Motion Compensation Precision" (AMCP) [5] [6] and "Adaptive Color Transform" (ACT) [7].

Recognizing the market need and SCC potentials, many industrial companies are currently following this incoming extension during standardization and possibly may include these new coding techniques into their products. How to efficiently bridge the existing HEVC and the incoming HEVC-SCC extension through video transcoding (VTC) becomes interesting and useful, especially during the phase when baseline HEVC and novel HEVC-SCC coexist.

VTC is a useful technology to realize video adaptation. It converts incoming bitstream from one version to another. Many properties may change during transcoding, such as video format, video bitrate, frame rate, spatial resolution and coding tools used. In the literature, the conversion within the same format (e.g., spatial rescaling within H.264) is usually referred as "homogeneous transcoding" while the conversion between different formats (e.g., between H.264 and HEVC) is referred as "heterogeneous transcoding". Beyond that, even additional information could be inserted during transcoding, such as watermarking, error resilience, etc. From the commercial product perspectives, a central server is used to periodically examine clients' constraints (e.g., bandwidth) and tailor suitable bit-streams accordingly, as shown in Fig. 1. Though it is possible to use the "trivial" approach, which firstly decodes the source bitstream and then completely re-encodes into target bitstream, however, such approach proves inefficient from complexity point of view. A reasonable approach is to fully utilize decoded side information from the source stream to assist re-encoding such that coding performance is preserved while the re-encoding speed is significantly improved.

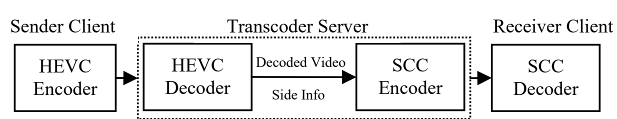

Figure 1. HEVC-SCC Transcoding Framework

There are many pioneer works in VTC area. In [8], a high-level video transcoding overview is presented from an architecture perspective, focusing on spatial and temporal resolution reduction, DCT-domain down-conversion, etc. When HEVC is introduced, a huge amount of VTC studies were redirected into "H.264-HEVC" transcoding area. For instance, Peixoto, et al. proposed several machine learning and statistics based frameworks (e.g., [9] [10] [11]) to improve HEVC re-encoding speed. In their papers, H.264 Macroblocks (MBs) are mapped into HEVC coding units (CU) based on motion vectors (MVs) distribution through either online or offline training. Incorporated with statistics-based fast termination criteria, the proposed schemes could introduce an over 3x encoder speedup with a 4% RD loss compared with trivial transcoder. Diaz-Honrubia, et al. also proposed a series of fast VTC schemes (e.g., [12] [13]) to exploit H.264/AVC decoded side information for HEVC CU partition decision based on a Naïve-Bayes (NB) classifier, specifically for CUs with size 32x32 and 64x64, whereas for smaller CUs, the proposed transcoder mimics H.264/AVC behaviors. A quantitative speed-up of about 2.5x is reported with only a 5% BD-Rate penalty. In [14], a HEVC fast transcoder is proposed based on merged block prediction homogeneity. Residuals and MV consistencies are populated to represent the homogeneity of target region and decide CU partition. Even though there are a few non-mainstream VTC frameworks, such as CU classification into background and foreground blocks based on scene modeling for surveillance applications (e.g., [15]), however, the major VTC research directions are still dominated by mode mapping and machine learning techniques.

Beyond these prior works, to our best knowledge, we are the first group addressing HEVC-SCC transcoding based on official SCM software (which is mostly likely to be adopted and deployed into the next-generation screen content products). Within the scope of this paper, we concentrate on the heterogeneous transcoding from baseline HEVC to HEVC-SCC extension for Intra-frame coding, to efficiently determine the usages of new SCC tools.

The sequel of this paper is structured as follows. Section 2 briefly reviews SCM software architecture, new coding tools and major challenges for SCC fast transcoding. In Section 3, our proposed fast SCC transcoding framework is illustrated and described in more details. In Section 4, experimental results are presented. This paper concludes in Section 5 with some future work summarized.

2. SCREEN CONTENT MODEL (SCM) - REVIEW

SCM is the JCTVC official test model software for SCC extension development. Beyond HEVC, new tools are introduced to improve the coding efficiency.

2.1. SCM Mode and Partition Decisions

SCM shares exactly the same flexible quadtree block partitioning scheme as HEVC, which enables the use of CUs, Prediction Units (PUs) and Transform Units (TUs) to adapt to diverse picture contents. CU is the basic unit for mode decision and is always in square shape. The Coding Tree Unit (CTU), is of 64x64 pixels by default. At encoder, incoming pictures are divided into non-overlapping CTUs and each CTU can be further divided into four equal-sized smaller CUs recursively, until the maximum depth is reached, as shown in Fig. 2. At each CU-level, to determine the optimal encoding parameters (e.g.: partition decision, mode decision, etc.), an exhaustive search is employed by comparing RD costs using different coding modes and comparing the minimum RD cost at current CU level against the sum of RD costs of its sub-CUs (each using best mode and partition).

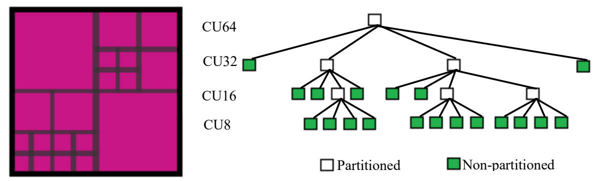

Figure 2. SCM CU Hierachitical Quadtree Partitioning Structure

2.2. SCM New Coding Tools beyond HEVC

SCM adopts two major coding tools.

Intra Block Copy [2] [3] is an Intra-frame version of the block matching scheme. To compress the current CU, the encoder will look back into previously-coded area and find the best matching block. If chosen, a "Block Vector" (BV) will be signaled to indicate the spatial offset between the best matching block and the current CU.

Palette Mode [4] encodes current CU as a combination of a color table and an index map. Color table stores representative color "triplets". The original pixel block is then translated into the corresponding index map indicating which color entry in the table is used for each pixel location.

Besides, there are several other coding options, such as adaptive color transform [7], adaptive motion compensation precision [5] [6], etc. Given the space limit and the coverage of this paper, such coding options will not be discussed.

2.3. HEVC-SCC Fast Transcoding Challenges

Unlike conventional HEVC Intra-modes, SCC modes are highly dependent on the repetitive graphical patterns and image colors that previously appeared. This "historical dependency" makes fast partition decision and VTC mode mapping much more complicated and challenging. For IBC blocks, depending on whether similar pattern appeared previously, encoding costs of same CU pattern but at different locations may vary significantly. Similarly, for PLT coding mode, two color tables are maintained. One is used for current CU and the other (also referred as "palette predictor") is used as a dynamic lookup table storing the historical colors used. Depending on whether similar colors appeared before and how frequent these colors are, the PLT coding costs of the same CU pattern but at a different location may also vary significantly. Furthermore, these new coding modes and options allow "inhomogeneous" blocks to be encoded as a larger block without splitting. As shown in Fig. 3, 16x16 textual CUs in the top row are encoded using PLT mode (in green) directly without splitting into smaller 8x8 Intra CUs (in purple) in the bottom row.

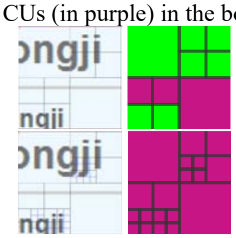

Figure 3. CTU Partition Decision Comparison between SCM and HEVC (Top: Text CTU coded by SCM-4.0; Bottom: Text CTU coded by HEVC)

To conclude, due to the unique signal characteristics of screen contents and the designs of PLT and IBC algorithms, existing VTC mode mapping and fast splitting termination approaches cannot be directly applied onto SCM. How to accurately and efficiently map HEVC modes and partitions onto SCC modes and decide the optimal SCC partition is a challenging problem, even for human judgment.

3. PROPOSED FAST HEVC-SCC TRANSCODER FOR INTRA-FRAME CODING

We have some valuable prior knowledge about the coding behaviors of HEVC and SCC, as summarized below. Such information guides us through our transcoder design.

3-A. For flat, smoothly-varying or directional blocks (e.g., in Fig. 4), HEVC and SCC will both use Intra-mode without further partitions. The transcoder may simply copy the Intra mode from HEVC and apply to SCC in these cases.

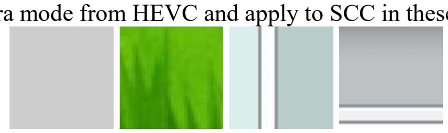

Figure 4. Sample Blocks Coded by Intra Mode in SCM (Left to Right: Flat, Smoothly-Varying, Vertical and Horizontal patterns)

3-B. SCC new coding modes enable "inhomogeneous" blocks to be encoded in larger CUs. As shown in Fig. 3, compared with Intra-mode, PLT and IBC modes tend to be chosen at larger CU size. An intuitive yet safe transcoding heuristic is that CU hierarchical depth of SCC should be shallower than the depth of HEVC. For example, in Fig. 3, RD-optimal coding depth is 3 using SCC but 4 using HEVC.

3-C. Computer-generated area (such as icon, graphics, etc.) is usually coded using PLT or IBC. Camera-captured area (e.g., picture) is typically coded using Intra. A fast and accurate classification between Screen Content Block (SCB) and Natural Image Block (NIB) may significantly reduce the RDO mode candidates from {Intra, PLT, IBC} to either {Intra} or {PLT, IBC}. Both decoded side information and CU statistical features may be jointly analyzed for such categorization. For instance, SCBs tpyically contain fewer distinct color number and sharper edges.

3-D. Though HEVC will split large SCBs into smaller Intra CUs, the final residual image after partitioning is sparse, while for NIBs, the residual image is less sparse, as shown in Fig. 5 from the right columns, where SCB residual (after HEVC Intra prediction) has only 711 nonzero pixels while NIB residual has 3273 nonzero pixels.

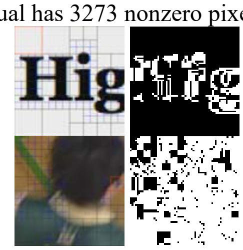

Figure 5. Screen Content and Natural Content Intra Residual Analysis (Top Row: SCB sample; Bottom Row: NIB sample; Left Column: Image Block; Right Column: Residual Map w/ white pixels as nonzero entries)

According to such statistical knowledge, our proposed transcoder is designed and implemented as a pre-analysis module before Intra-frame mode selection, shown in Fig. 6.

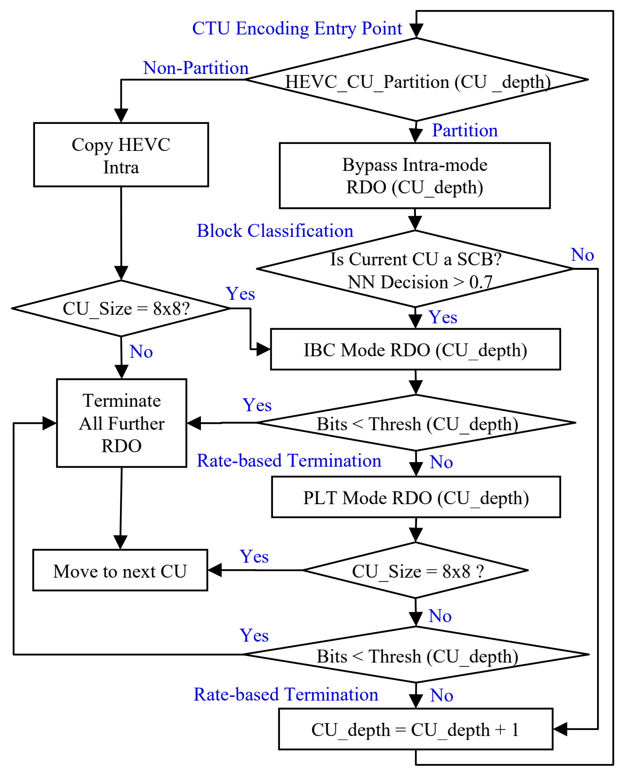

Figure 6. Proposed Fast HEVC-SCC Transcoding Workflow

To determine whether the current CU is a possible SCB, we design a classifier based on neural network (NN) using features 1 through 3 from our previous work [16] and new feature 4 motivated by our discussion in Section 3-D.

Feature 1: CU Variance;

Feature 2: CU Distinct Color Number;

Feature 3: CU Gradient Kurtosis;

Feature 4: CU HEVC Intra Residual sparsity, defined as the L-0 norm of residual image matrix.

Our training and validation set (TVS) contains totally 5,940 CU64 samples, 23,760 CU32 samples, 96,480 CU16 samples from three SCC testing sequences (i.e., "Console", "Desktop", "FlyingGraphics"). Since that our preliminary results show IBC mode utilization at CU8 level is mostly determined by global search rather than local statistics, we do not use block classification at this level to better preserve coding performance. For simplicity, we randomly choose half samples as training set (TRS) and the other half as validation set (VLS).

Our ground-truth decision block labels are obtained by re-encoding decoded HEVC videos according to SCC common testing conditions in [17] using SCM-4.0. If the entire block is encoded using Intra mode, regardless of partition condition, the block is labeled as an "NIB". If the block is encoded using SCC mode, either purely by SCC or mixed with Intra, the block label is labeled as an "SCB".

Block classifiers on each CU size are trained with a 2-layer NN structure with two "sigmoid" transfer functions between input and hidden layer and between hidden layer and output. Given that we have sufficient samples, cross-validation is assumed unnecessary and not used. Instead, we tune the optimal hidden layer node numbers directly according to the prediction accuracies on VLS. In our models, the optimal hidden node numbers are 2, 3 and 3 for CU64, CU32 and CU16, respectively. The NN training is implemented using MATLAB NN Toolbox (Ver. 8.1).

The NN classifiers output a soft-decision between 0 and 1. A block with a decision value closer to 1 has a higher probability to be a NIB. Similarly, a block with a decision value closer to 0 has a higher probability to be an SCB. To preserve coding performance, we apply a biased decision boundary value of 0.7. Namely, we classify a block as NIB only if the NN decision is greater than 0.7 and bypass SCC modes at corresponding CU levels. The boundary selection provides a tunable tradeoff between coding performance and complexity saving. A smaller decision boundary setting will classify more blocks into NIBs and increase the complexity reduction but degrade coding efficiency.

To verify the model generalization accuracies, we apply our trained classifiers on four unseen SCC sequences (including "Programming", "SlideShow", "WebBrowsing" and "BasketballScreen") in testing set (TSS) and report the performances in Table I.

The rate-based fast termination thresholds in Fig. 6 are determined based on coding bits consumed at each CU level and are chosen empirically and conservatively to better preserve coding performance. The values we use are 20, 12 and 4 for CU32, CU16 and CU8 levels, respectively.

4. EXPERIMENTAL RESULTS

Our proposed HEVC-SCC fast transcoding framework is evaluated as follows: Firstly, 7 official SCC sequences (with sample frames in Fig. 7) are encoded using anchor HEVC (Ver. 16.4) according to common testing conditions (CTC) with 4 QPs (i.e., 22, 27, 32 and 37). Later, HEVC bitstream is decoded into YUV distorted videos with side information retrieved and stored. Finally, our proposed transcoder will load and analyze distorted videos and side information and re-encode distorted videos using same QPs.

Figure 7. Sample Frames from JCT-VC SCC Sequences

The SCC re-encoding performances are evaluated using homogeneous Windows 7 (64-bit) desktops with Intel-i5 CPU (2.67 GHz dual cores) and 4GB RAM. The complexity reduction is directly measured by encoding time. Compared with anchor SCM-4.0, our proposed scheme can achieve a 48% re-encoding complexity reduction with only 2.0% BD-Rate [18] loss. Simulation results are provided in Table I.

TABLE I. PROPOSED TRANSCODER PERFORMANCE EVALUATION

Sequence	QP	*Anchor SCM-4.0*			*Proposed*			*Performance*	
		Rate	*PSNR*	*Time*	*Rate*	*PSNR*	*Time*	*Rate*	*Time*
Programming+ 720p YUV	22	542536	49.62	43.2	549720	49.64	26.01	+1.05%	-42.9%
	27	386224	45.42	39.7	389408	45.31	22.63		
	32	267264	40.79	34.9	268360	40.85	19.46		
	37	192944	36.44	30.9	197568	36.45	17.13		
SlideShow+ 720p YUV	22	373184	51.78	33.4	378696	51.66	16.50	+2.21%	-51.4%
	27	268352	47.95	30.2	270584	47.80	15.20		
	32	195680	44.46	26.7	197320	44.30	13.25		
	37	134056	40.48	24.6	134576	40.36	11.08		
WebBrowsing+ 720p YUV	22	286456	52.53	38.7	290520	52.58	19.00	+3.08%	-47.9%
	27	232328	48.03	32.9	236392	47.80	17.89		
	32	169048	44.23	30.3	171344	43.93	16.29		
	37	127440	38.05	26.5	130504	37.77	13.58		
Basketball+ 1440p YUV	22	127013	52.14	210.5	131481	51.93	113.2	+3.13%	-46.1%
	27	85529	48.27	168.2	86322	48.19	87.1		
	32	59184	45.26	143.9	60575	45.13	77.3		
	37	39551	41.23	125.6	39939	41.11	70.8		
Console* 1080p YUV	22	627400	51.79	83.8	640960	51.95	43.91	+1.85%	-49.0%
	27	558648	47.25	82.7	569968	47.17	41.10		
	32	465992	42.97	77.8	473880	43.01	38.14		
	37	358304	38.47	68.9	363968	38.48	36.48		
Desktop* 1080p YUV	22	708400	50.59	116.9	719056	50.57	61.50	+1.72%	-48.1%
	27	637944	46.13	114.4	646584	46.27	59.79		
	32	599872	40.30	109.8	611928	40.43	55.84		
	37	522744	35.40	98.9	539160	35.34	51.29		
FlyingGraphics* 1080p YUV	22	1431048	48.75	118.9	1467784	48.74	61.68	+1.94%	-50.1%
	27	1105376	44.30	109.2	1130208	44.31	53.54		
	32	824000	40.32	98.8	829944	40.24	48.54		
	37	550840	35.95	85.1	559640	35.90	42.02		

(Rate: Bitrate in kbps; PSNR: Y component PSNR in dB; Time: frame encoding time in second)
(Sequences* are used for training and validation; Sequence+ are used for testing and verification)

5. CONCLUSIONS

In this paper, a novel HEVC-SCC transcoding scheme is presented based on screen content statistical study and machine learning. On top of SCM-4.0, our proposed scheme can introduce a 48% complexity reduction on average with only 2.14% BD-Rate increase. Future study of this project may generalize our current framework to SCC Inter-frame transcoding.

6. REFERENCESS

[1] G.-J. Sullivan, J. Boyce, Y. Chen, J.-R. Ohm, A. Segall, and A. Vetro, “Standardized Extensions of High Efficiency Video Coding (HEVC),” IEEE Journal of Selected Topics in Signal Processing, vol. 7, no. 6, pp. 1001 – 1016, Dec. 2013.

[2] C. Pang, J. Sole, L. Guo, M. Karczewicz, and R. Joshi, “Non-RCE3: Intra Motion Compensation with 2-D MVs,” Doc. JCTVC-N0256, July 2013.

[3] J. Chen, Y. Chen, T. Hsieh, R. Joshi, M. Karczewicz, W.-S. Kim, X. Li, C. Pang, W. Pu, K. Rapaka, J. Sole, L. Zhang, and F. Zou, “Description of screen content coding technology proposal by Qualcomm,” Doc. JCTVC-Q0031, April 2014.

[4] L. Guo, M. Karczewicz, and J. Sole, “RCE3: Results of Test 3.1 on Palette Mode for Screen Content Coding,” Doc. JCTVC-N0247, July 2013.

[5] X. Li, J. Sole, M. Karczewicz, “Adaptive MV precision for Screen Content Coding,” Doc. JCTVC-P0283, January 2014.

[6] B. Li, J. Xu, G. J. Sullivan, Y. Zhou, B. Lin, “Adaptive motion vector resolution for screen content,” Doc. JCTVC-S0085, October 2014.

[7] L. Zhang, J. Chen, J. Sole, M. Karczewicz, X. Xiu, Y. He, Y. Ye, “SCCE5 Test 3.2.1: In-loop Color-Space Transform,” Doc. JCTVC-R0147, June 2014.

[8] A. Vetro, C. Christopoulos, and H.Sun, “Video Transcoding Architectures and Techniques: An Overview”, IEEE Signal Processing Magazine, vol. 20, no. 2, pp.18-29. March 2003.

[9] E. Peixoto, B. Macchiavello, R. Queiroz, E.M. Hung, “Fast H.264/AVC to HEVC Transcoding Based on Machine Learning,” International Telecommunications Symposium (ITS), pp. 1-4, 2014.

[10] E. Peixoto, B. Macchiavello, E.M. Hung, A. Zaghetto, T. Shanableh, and E. Lzquierdo, “An H.264/AVC to HEVC video transcoder based on mode mapping,” IEEE International Conference on Image Processing (ICIP), pp. 1972-1976, September 2013.

[11] E. Peixoto, B. Macchiavello, E.M. Hung, and R. Queiroz, “A fast HEVC transcoder based on content modeling and early termination,” IEEE International Conference on Image Processing (ICIP), pp. 2532-2536, September 2014.

[12] A. J. Diaz-Honrubia, J. L. Martinez, J. M. Puerta, J. A. Gamez, J. De Cock, and P. Cuenca, “Fast Quadtree Level Decision Algorithm for H.264/HEVC Transcoder,” in IEEE International Conference on Image Processing (ICIP), pp. 2497–2501, October 2014.

[13] A. J. Diaz-Honrubia, J. L. Martinez, J. M. Puerta, J. A. Gamez, J. De Cock, and P. Cuenca, “A Data-Driven Probabilistic CTU Splitting Algorithm for Fast H.264/HEVC Video Transcoding,” in Data Compression Conference (DCC), pp. 449, April 2015.

[14] F. Zheng, Z. Shi, X. Zhang and Z. Gao, “Effective H.264/AVC to HEVC transcoder based on Prediction Homogeneity,” in IEEE Visual Communications and Image Processing Conference (VCIP), pp. 233-236, December 2014.

[15] P. Xinga, Y. Tian, X. Zhang, Y. Wang, and T. Huang, “A Coding Unit Classification based AVC to HEVC transcoding with background modeling for surveillance videos,” in IEEE International Conference of Visual Communication and Image Processing (VCIP), pp. 1-6, 2013.

[16] F. Duanmu, Z. Ma, and Y. Wang, “Fast CU partition decision using machine learning for screen content compression”, in IEEE International Conference on Image Processing (ICIP), pp. 4972 - 4976, September 2015.

[17] H. Yu, R. Cohen, K. Rapaka, J. Xu, “Common Test Conditions for Screen Content Coding,” Doc. JCTVC-T1015, February 2015.

[18] G. Bjontegaard, “Calculation of Average PSNR differences Between RD Curves (VCEG-M33),” in VCEG Meeting (ITU-T SG16 Q.6), Austin, Texas, USA, Apr. 2-4, 2001.