The following publication W. Kuang, Y. -L. Chan, S. -H. Tsang and W. -C. Siu, "Online-Learning-Based Bayesian Decision Rule for Fast Intra Mode and CU Partitioning Algorithm in HEVC Screen Content Coding," in IEEE Transactions on Image Processing, vol. 29, pp. 170-185, 2020 is available at https://doi.org/10.1109/TIP.2019.2924810.

Online-Learning-Based Bayesian Decision Rule for Fast Intra Mode and CU Partitioning Algorithm in HEVC Screen Content Coding

Wei Kuang, *Student Member*, *IEEE*, Yui-Lam Chan, *Member*, *IEEE*, Sik-Ho Tsang, *Member*, *IEEE*, and Wan-Chi Siu, *Life Fellow*, *IEEE*

***Abstract*—Screen Content Coding (SCC) is an extension of High Efficiency Video Coding by adopting new coding modes to improve the coding efficiency of SCC at the expense of increased complexity. This paper proposes an online-learning approach for fast mode decision and coding unit (CU) size decision in SCC. To make fast mode decision, the corner point is firstly extracted as a unique feature in screen content, which is an essential pre-processing step to guide Bayesian decision modelling. Second, distinct color number in a CU is derived as another unique feature in screen content to build the precise model using online-learning for skipping unnecessary modes. Third, the correlation of the modes among spatial neighboring CUs is analyzed to further eliminate unnecessary mode candidates. Finally, the Bayesian decision rule using online-learning is applied again to make fast CU size decision. To ensure the accuracy of the Bayesian decision models, new scene change detection is designed to update the models. Results show that the proposed algorithm achieves 36.69% encoding time reduction with 1.08% Bjøntegaard delta bitrate (BDBR) increment under All Intra configuration. By integrating into the existing fast SCC approach, the proposed algorithm reduces 48.83% encoding time with 1.78% increase in BDBR.**

***Index Terms*—Screen Content Coding (SCC), High Efficiency Video Coding (HEVC), fast mode decision, fast CU size decision, Bayesian decision rule, scene change detection.**

I. Introduction

WITH recent fast development of the Internet and wireless communication, screen content coding (SCC) has been developed for many video applications, such as desktop sharing, cloud computing, and web conferencing. Unlike camera-captured videos with only natural image blocks (NIBs), screen content videos also contain screen content blocks (SCBs), which have no noisy, many strong corners, a limited number of different colors, and many identical blocks within a frame. High Efficiency Video Coding (HEVC) is designed for NIBs in camera-captured content, but it cannot compress SCBs in screen content videos efficiently. Therefore, SCC [1] has been included in the HEVC standard [2] as one of its extensions, and two new coding modes: intra block copy (IBC) [3] and palette (PLT) [4] have been added to improve the coding performance of SCC but induce intensive computational complexity.

HEVC adopts the flexible coding tree unit (CTU) partitioning structure to improve the coding efficiency, but it leads to 220% computational complexity increase compared with the previous H.264/AVC standard [5] in the All Intra (AI) case [6]. In the HEVC-based SCC, an encoder additionally checks the new IBC and PLT modes in addition to the conventional intra (Cintra) mode for a coding unit (CU), and the new modes cause a further surge in the computation complexity. For the SCC reference software – Screen Content Coding Test Model version 7.0 (SCM-7.0), about 61% of the computational complexity in the mode searching process is brought by IBC and PLT modes. Therefore, reducing the complexity of SCC is essential for computation and energy constrained applications.

Recently, many approaches have been proposed to speed up the encoding process of HEVC. Those efforts are mainly divided into fast CU partition [7-14], and fast Cintra directional mode decision [15], [16]. Specifically, good performances are provided in [9-11], where the Bayesian decision rule is utilized to make early CU split and pruning decisions for NIBs. However, the new IBC and PLT modes make CU size decision of SCC different from HEVC, where an inhomogeneous CU is possible to be encoded as a large block without partitioning. Therefore, the algorithms in [9-11] are ineffective when applied to SCC. Besides, the new IBC and PLT modes make the fast mode decision of SCC much more challenging. These new modes make all fast HEVC algorithms in [7-16] fail in fast mode decision of Cintra, IBC and PLT, as they only consider the characteristics of NIBs without the newly introduced IBC and PLT modes.

To reduce the computational complexity of SCC, existing SCC fast algorithms can be divided into IBC searching algorithms [17-20], fast CU partition algorithms [21-23], and fast algorithms where mode decision and CU partition decision are all considered [24-26]. To reduce the computational complexity of IBC mode in SCC, the works in [17-19] utilize features such as the rate-distortion (RD) cost of Cintra mode,

Manuscript received December 21, 2017. This work was supported by the Center for Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, the research studentship provided by the University, and a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Grant No. PolyU 152530/16E).

The authors are with the Center for Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: wei.kuang@connect.polyu.hk; enylchan@polyu.edu.hk; sik-ho.tsang@polyu.edu.hk; enwcsiu@polyu.edu.hk).

hash value, CU activity and gradient to skip unnecessary IBC checking. In [20], a new mode was designed to fill a noiseless smooth CU by its boundary samples.

To speed up the CU partition process of SCC, conventional neural network-based classifiers are trained to make fast CU size decision by utilizing features that describe CU statistics and sub-CU homogeneity in [21]. However, it induces high RD performance loss. In [22], the depth information of the collocated CU is used to predict the depth level of the current CU. However, this method can only handle the stationary CUs well. In [23], by using rules based on CU entropy and coding bits, early termination is made for CU partitions adaptively.

To make both fast mode decision and fast CU size decision of SCC, CU content classifications were conducted in [24-27]. However, they mainly focus on the fast mode decision for traditional NIBs by CU type classification. In [24], decision tree-based classifiers are used to classify incoming blocks into screen content blocks SCBs and NIBs. Two classifiers are specially designed for NIBs, and they predict Cintra directional mode and terminate CU partition of NIBs, respectively. For SCBs, only the Cintra mode is skipped so that both IBC and PLT modes need to be checked. Although thresholds are set to skip remaining modes and CU partitions, it is only useful for CUs with small encoding bits. Similarly, the work in [25] also classifies CU into NIBs and SCBs by analyzing content characteristics. IBC and PLT mode candidates are skipped for NIBs but all modes are checked for SCBs. Besides, bit per pixel in the current CU and the neighboring/collocated CU depth information are used to make fast CU size decision. In [26], Intra mode is checked for all CUs with 2N×2N prediction units (PUs) to collect features. Then decision tree-based classifiers are invoked to make CU type decision and CU partitioning decision. If a CU is classified as a SCB, both IBC and PLT modes are checked. Otherwise, only Cintra mode is checked for N×N PUs in the depth level of 3 for NIBs. In [27], conventional neural network-based classifiers are trained to classify CUs in NIBs and SCBs. Again, IBC, PLT modes and a subset of Cintra mode are checked for SCBs, while only Cintra mode is checked for NIBs. Then, information from spatial and temporal adjacent CUs is utilized to early terminate CU patriation.

Compared with pre-trained models and pre-tuned heuristics in [17-27], online-learning has the advantage of generating decision models adaptively according to the content being encoded. Once the framework of the fast algorithm has been decided, it can be directly applied to sequences with different content characteristics, QPs and color formats without any modification because the decision model is always updated according to the content being encoded. On the contrary, pre-trained models or pre-tuned heuristics are derived with limited training data and then they are applied to all sequences. Therefore, they usually have the generalization problem if the testing sequences have different characteristics from the training data, such as different content characteristics, QPs or color formats. In this paper, we therefore propose an online-learning-based fast mode decision and CU size decision approaches using the Bayesian decision rule. Bayesian decision rule is friendly to online-learning since it only needs to estimate the priori probability and likelihood function when there is a new scene, and then the derived model is applied to the following frames to make fast decision. Therefore, it costs much less time for model training than other classifiers such as decision trees or neural networks. In brief, the main contributions of this paper are summarized as follows.

- Bayesian decision rule is one of the well-known classification tools used in video coding. As a classification problem, some representative features such as Rate-Distortion (RD) cost, variance of prediction errors, etc. in camera-capture contents are always used for training [9-11]. However, these features cannot characterize screen contents. Corner point (CP) is a unique feature to characterize screen contents since SCBs usually contain sharp corners, while NIBs do not. In this paper, CP is explored to divide CUs into two groups with and without CP. We find that these two groups cannot share the same Bayesian decision model. One of the contributions in this paper is how to properly adopt CPs in the Bayesian decision rule for SCC. We reveal that CP is an excellent complement of the Bayesian decision rule for mode decision in SCC
- Distinct color number in a CU is derived as another unique feature in screen content sequences, and it is used to build two Bayesian decision models for CU with and without CPs using online-learning for skipping unnecessary modes.
- The correlation of the modes among spatial neighboring CUs is analyzed to further eliminate mode candidates of the current CU.
- For CU size decision, CUs with the same optimal modes are grouped together to build Bayesian decision models using online-learning for early terminating unnecessary partitions.
- A new low-complexity scene change detection method is specifically designed for screen contents to update statistical parameters adaptively for our proposed online-learning SCC algorithm in different scenes.

The differences between our contributions and the related schemes can be summarized as follows.

- The new IBC and PLT modes make CU size decision of SCC very different from HEVC, so that the fast CU size decision using the Bayesian decision rule in [9-11] cannot be efficiently applied to SCC. Besides, these Bayesian decision rule in [9-11] are only limited to CU size decision, but not used for fast mode decision in SCC since there is only Cintra mode in HEVC. The introductions of IBC and PLT make the necessity of a completely new fast mode decision method in SCC with the help of new features. To the best of our knowledge, we are the first to use the Bayesian decision rule for the fast mode decision by analyzing the characteristics of CUs with various modes including both newly introduced IBC and PLT modes.
- Unlike the algorithms in [17-20], which only simplify IBC mode of SCC, and the algorithms in [21-23], which only simplify CU size decision of SCC, we consider the whole encoding process of SCC to provide more encoding time reduction.
- Although the algorithms in [24-27] can speed up both mode decision and CU size decision of SCC, the fast mode decision in [24-27] is from the idea of CU type decision. They treat the decisions for IBC and PLT modes the same so that at least two modes (IBC+PLT or Cintra+IBC+PLT) are checked for a SCB. Comparatively, the proposed fast

mode decision techniques make decision for each mode separately, so that many SCBs can only check one mode;

- Unlike [17-27] applying pre-trained models and pre-tuned heuristics to all sequences, the proposed algorithm generates decision models adaptively according to the content being encoded by using online-learning.
- Different from [22] [25] and [27], our CU size algorithm does not need information from the collocated CU. When it is feasible to make CU depth prediction using the mode and CU depth information from the collocated CU, the proposed CU size decision can also work with it to further speed up the SCC encoder.

The rest of the paper is organized as follows. Section II briefly reviews and analyzes the original mode decision scheme in SCC. Section III presents the proposed fast mode and CU size decision techniques. Section IV gives the experimental results and discussions. Finally, Section V concludes this paper.

II. Review and Analysis on Mode Decisions in SCC

Due to different characteristics in screen content videos, IBC mode and PLT mode were proposed for SCC as additional coding tools. IBC mode is a block matching-based approach which can be considered as motion compensation within the current reconstructed frame. The syntax for IBC mode is unified with inter-mode but it adopts different searching strategy from inter-mode. IBC mode is firstly checked using block vector (BV) predictors of skip and merge modes, which is an intra version of the skip and merge modes in inter-prediction. Prediction residual is signaled to the decoder for merge mode, while it is omitted for skip mode to reduce bitrate. If skip mode is chosen as the best mode so far, the mode search process of a CU is early terminated and the following IBC search and PLT mode are skipped. Otherwise, IBC search is performed for CUs with sizes of 16×16 and 8×8, which is an intra version of the advanced motion vector prediction (AMVP) mode in inter-prediction. For 16×16 CUs, only 2N×2N PUs are checked while PU partitions of N×2N and 2N×N are also allowed for 8×8 CUs. Besides, a new hash-based search is enabled for 8×8 CUs with 2N×2N PUs, and only blocks with the same hash value as the current CU are checked. Finally, the optimal BV is signaled using the syntax of AMVP mode. PLT mode is another effective approach applied for CU sizes from 32×32 down to 8×8. The idea of PLT mode comes from the observation that SCBs often contain a limited number of sample values. Several representative sample values in a CU are selected as base colors to form a palette table, and an index map is generated to send color indices for each position. A detailed technical overview of IBC and PLT can be found in [1] and [3].

SCC inherits the same coding structure from HEVC. Each CTU can be recursively partitioned into four sub-CUs until the smallest coding unit (SCU) size of 8×8 is reached. In SCC intra coding, three modes, Cintra mode, IBC mode and PLT mode, will both be checked for all depth levels, and then the optimal CU partition with the optimal mode will be selected by comparing their RD costs. However, this decision process leads to high computational complexity of the encoder. Fig. 1 shows the mode selection results for "WebBrowsing" with the quantization parameter (QP) of 22 and the depth level of 2. As IBC and PLT modes are specially designed for SCBs, they have higher selection probabilities for SCBs, while NIBs tend to select Cintra mode because of fewer repeated patterns and high number of sample values. It is noted that single color CUs can be encoded efficiently by all modes, thus they may select any mode as their optimal mode. Besides, due to the introduction of new IBC and PLT modes, SCBs with complex texture may also select large size CUs. Therefore, new fast mode decision and CU size decision methods based on these different characteristics between NIBs and SCBs are highly desired.

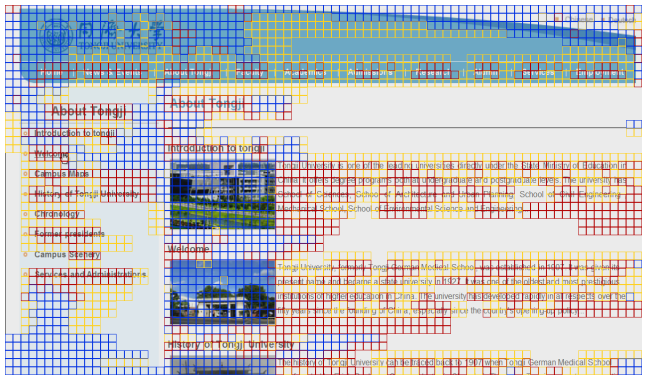


Fig. 1. Mode distribution for "WebBrowsing" with QP of 22 and the depth level of 2. Cintra, IBC and PLT modes are denoted by blue, yellow and red blocks, respectively. Those regions without any colors denoted are terminated at depth levels of 0 or 1.

III. Proposed Fast Mode and CU Size Decision Algorithm

In SCC, the computational complexity mainly stems from the RD cost computation of all modes in every depth level. Thus, it is very efficient to reduce the computational complexity if the mode and CU size decisions can be predicted precisely, and then all the remaining unnecessary RD cost computation can be skipped. As shown in Fig. 1, SCBs usually have strong corners and limited color numbers, which are more likely to select IBC and PLT modes. Meanwhile, NIBs are smoother and have higher color numbers, which are more likely to select Cintra mode. Besides, there exists spatial correlation among the current CU and its neighboring CUs, and the current CU is more likely to select the same optimal mode as that of its neighboring CUs. Furthermore, for a certain depth level, RD costs of unsplit CUs concentrate in the range with small values, while the RD costs of split CUs show relatively wide and flat distribution. By utilizing these observations, three techniques are designed in this paper for expediting mode decision and CU size decision in SCC, which are called fast mode decision by online-learning (FMD), mode decision refinement (MDR), and fast CU size decision by online-learning (FCUSD). Besides, to obtain the correct learning statistics for making decisions, a new scene change detection method, ratio of new distinct color number (RDN), is specifically designed for scene content which facilitates the proposed algorithm to update the learning statistics adaptively. The flowchart of the proposed fast algorithm is shown in Fig. 2, which will be explained in detail in following sub-sections.

A. Fast Mode Decision

The proposed fast mode decision algorithm determines whether early mode skip can be performed based on two techniques, FMD and MDR. FMD is made by using the online-

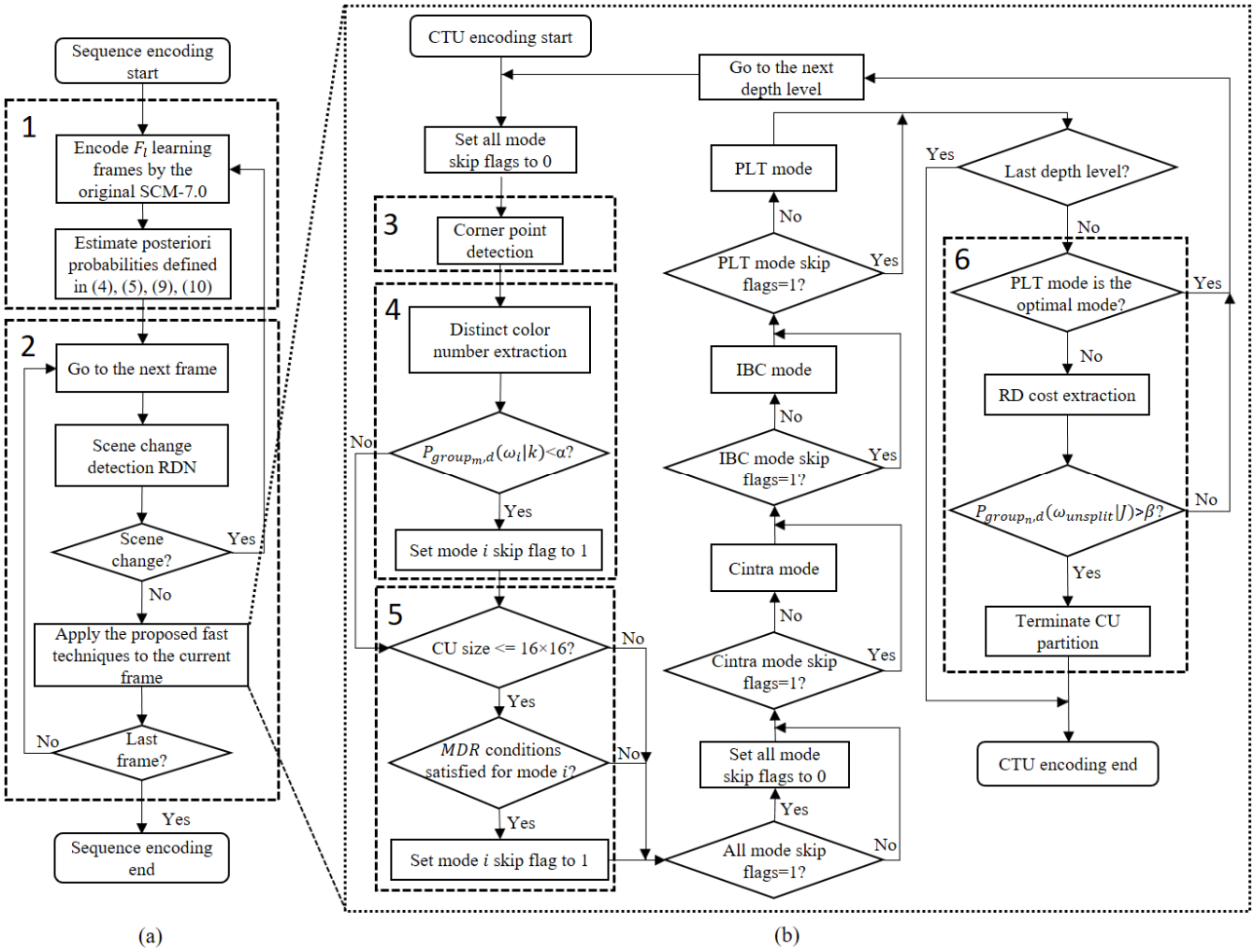


Fig. 2. Proposed fast mode and CU size decision flowchart. (a) Workflow of encoding a sequence, and (b) workflow of encoding a CTU. 1. Online-learning phase. 2. Proposed fast decision phase. 3. CP detection. 4. FMD. 5. MDR. 6. FCUSD.

learning based Bayesian decision rule, which early skips the modes with low probabilities for being optimal modes. Then, MDR utilizes the optimal mode information of spatial neighboring CUs to further reduce unnecessary mode candidates for the current CU. Since FMD and MDR consider the decision of each mode separately, the encoder can check one mode from IBC and PLT rather than always checking them together as in [24-27].

1) CP Detection

CP is a particular feature of SCBs which usually contain sharp corners while NIBs are smooth. Before utilizing the Bayesian decision rule to make fast mode decision, a CP detection is adopted as a pre-processing step shown in the module 3 of Fig. 2. The basic principle of CP detection is to find the interest points with two dominant and different edge directions in a local neighborhood of the point. Compared with NIBs, SCBs contain more strong corners. Thus, the Shi–Tomasi CP detection algorithm in [28] is applied to classify a frame into SCBs and NIBs roughly. In [28], an image with only luma component is used, which is denotes by I. The covariance matrix of an image patch at (u, v) and itself after shifted by (x, y) is written as

$$T(x,y) = \sum_u \sum_v w(u,v) \begin{bmatrix} {I_x}^2 & I_x I_y \\ I_x I_y & {I_y}^2 \end{bmatrix} \quad (1)$$

where $w(u, v)$ is a weight window. I_x and I_y are the partial derivatives of I. The eigenvectors of T are two principal directions and the eigenvalues of T reflect the degrees of the change in their directions. Thus, T should have two large eigenvalues θ_1 and θ_2, and the strength of a CP, ST, can be defined as

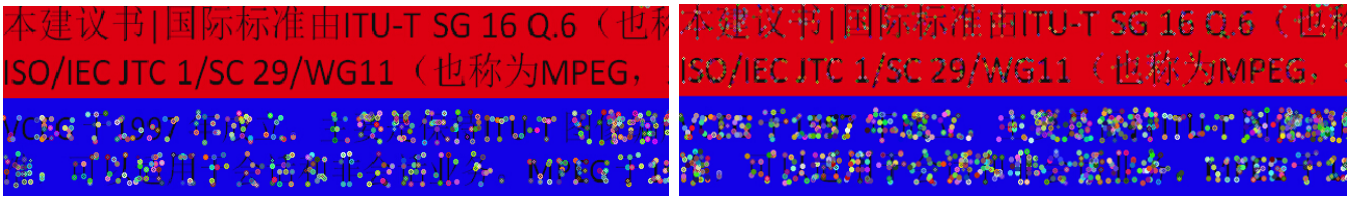
$$ST = min(\theta_1, \theta_2). \quad (2)$$

The minimal accepted strength ST_{min} of CPs in an image is determined as

$$ST_{min} = TH_{CP} \times ST_{max} \quad (3)$$

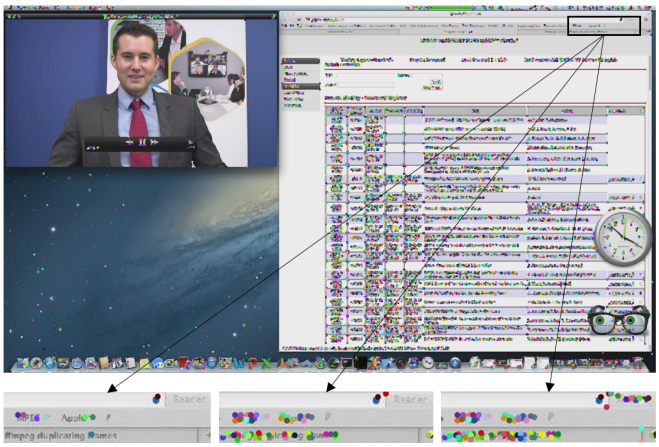
where ST_{max} is the largest strength of all CPs in the image, and TH_{CP} is a minimal accepted strength threshold. If the strength of a CP is larger than ST_{min}, this point can be detected as a CP. Otherwise, it is ignored.

We implemented the CP detection method based on



(a) CPs detected by luma component. (b) CPs detected by all components.

Fig. 3. Comparison of CP detection methods with (a) luma component (b) and all components (CPs are shown by color points).



(a) (b) (c)

Fig. 4. Detected CPs of "MissionControlClip2" with different values of TH_{CP}. (a) $TH_{CP} = 0.1$, (b) $TH_{CP} = 0.05$, and (c) $TH_{CP} = 0.01$ (CPs are represented by color points).

OpenCV2.4.9 function goodFeaturesToTrack(). For screen content videos, this method cannot detect CPs well with only luma samples used. Therefore, CPs in a frame are detected by considering all three components in our implementation. CPs are detected with each component using (1)-(3) separately, and then a pixel is treated as a CP if at least one of its components is detected as a CP. It is noted that this CP detection is very simple, and our experiment shows that it takes only 0.46% of the encoding time in the original SCM-7.0. CP detection with luma only and with all components are shown in Fig. 3(a) and (b), respectively, and better detection results can be obtained by using all components in Fig. 3(b). Fig. 4 shows the detected CPs of "MissionControlClip2" with different threshold values of TH_{CP}. As can be seen easily, SCBs contain many CPs while NIBs contain much less CPs. However, there are also some low contrast SCBs which are difficult to detect, as shown in the enlarged region of Fig. 4. Therefore, in our proposed algorithm, the strength of accepted CPs is set as a relatively small value with $TH_{CP} = 0.01$ to detect those low contrast CPs. Thus, CUs in a frame is divided into two groups now: CUs without corner CPs ($group_{NCP}$), and CUs with CPs ($group_{CP}$). Table I shows the mode distributions in $group_{NCP}$ and $group_{CP}$ for the first 100 frames of "MissionControlClip3" encoded with QP of 22. It is observed that most CUs in $group_{NCP}$ select Cintra as their optimal modes, while a significant number of CUs in $group_{CP}$ selects PLT or IBC as their optimal modes.

2) Fast Mode Decision by Online-learning (FMD)

After the pre-processing step of CP detection, early mode decisions are made for CUs in $group_{CP}$ and $group_{NCP}$, respectively. To build the Bayesian decision model for each

Table I

MODE DISTRIBUTIONS IN $group_{NCP}$ AND $group_{CP}$ FOR THE FIRST 100 FRAMES OF "MISSIONCONTROLCLIP3" ENCODED WITH QP OF 22.

$group_{NCP}$			
CU size	PLT	IBC	Cintra
64×64		3.37%	96.63%
32×32	3.03%	7.46%	89.51%
16×16	1.17%	24.79%	74.04%
8×8	0.90%	24.94%	74.16%
$group_{CP}$			
CU size	PLT	IBC	Cintra
64×64		25.15%	74.85%
32×32	65.71%	17.06%	17.23%
16×16	32.99%	49.01%	18.00%
8×8	12.45%	64.53%	23.02%

mode, the distinct color number of a CU is utilized as a unique feature in screen content sequences. To provide an accurate estimation for the distinct color number distribution, an online-learning based approach is adopted in our proposed algorithm. When encoding a video sequence, all statistical parameters of the Bayesian classifiers are obtained in the online-learning phase, where the first L frames of a new scene are encoded by the original SCM encoder to get these statistical parameters.

To make early mode decisions, three classes are defined as Cintra, IBC and PLT mode classes: ω_{Cintra}, ω_{IBC} and ω_{PLT} in each group. The distinct color number in a CU is extracted as the feature for classification. To fully utilize the pixel value information, a 24-bit sample value is utilized by concatenating the three components of a pixel. NIBs may have sensor noise while SCBs naturally concentrate on only a few sample values. Therefore, in each group, NIBs usually contain more distinct colors than SCBs. To analyze the statistical distribution, let k be the distinct color number in a CU varying from 1 to the total pixel number in this CU. $P_{group_{m},d}(k|\omega_i)$ is the conditional density of k in ω_i or the likelihood function, where $i\epsilon\{Cintra, IBC, PLT\}$, depth level $d\epsilon\{0,1,2,3\}$ and $m\epsilon\{NCP, CP\}$. $P_{group_{m},d}(k|\omega_i)$ can be obtained from the encoding statistics in the online-learning phase. To study the distributions of likelihood functions, $P_{group_{m},d}(k|\omega_i)$ of three typical sequences including "MissionControlClip3", "Programming" and "EBURainFruits", are shown in Figs. 5-7 with QPs of 22 and 37 at the depth level of 3, respectively. It is noted that "MissionControlClip3" and "Programming" contain both NIBs and SCBs while "EBURainFruits" only contains NIBs. As shown in Figs. 5-6(a) and (b), the likelihood distributions for different ω_i are similar, while diverse likelihood distributions in Figs. 5-6(c) and (d) are shown for different ω_i. In $group_{NCP}$, CUs in all classes show a concentrated distribution centered in the range with small values of k. However, in $group_{CP}$, most CUs in ω_{PLT} and ω_{IBC} contain a small value of k, while CUs in ω_{Cintra} tend to have a

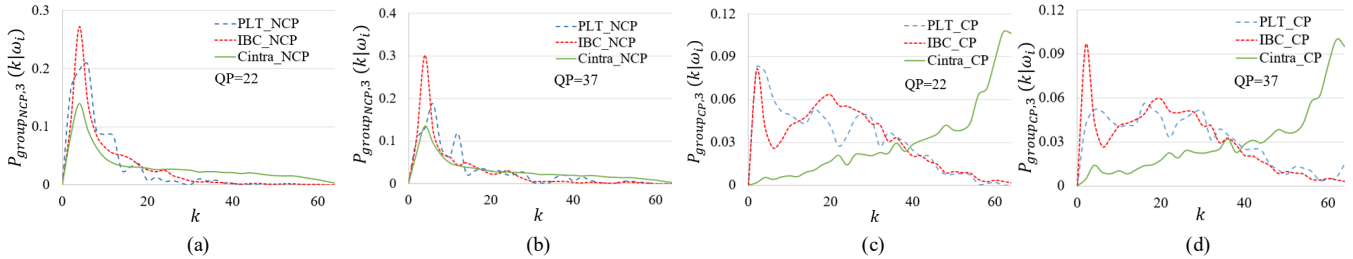


Fig. 5. Likelihood functions $P_{group_m,d}(k|\omega_i)$ for "MissionControlClip3" at the depth level of 3 in (a) $group_{NCP}$ with QP of 22, (b) $group_{NCP}$ with QP of 37, (c) $group_{CP}$ with QP of 22, and (d) $group_{CP}$ with QP of 37.

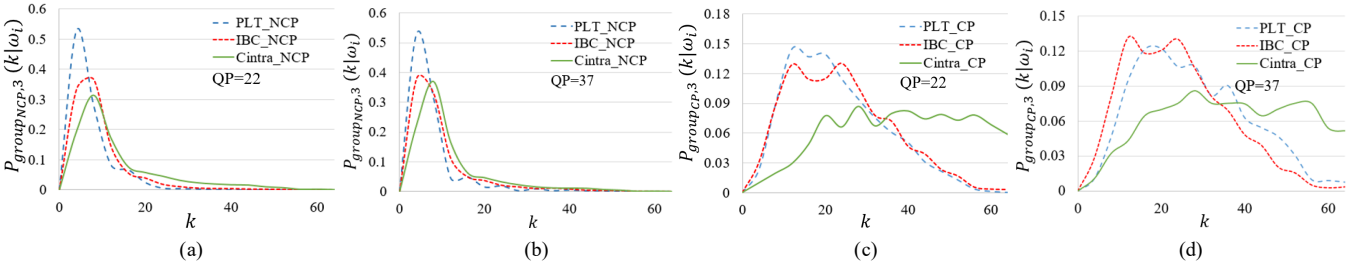


Fig. 6. Likelihood functions $P_{group_m,d}(k|\omega_i)$ for "Programming" at the depth level of 3 in (a) $group_{NCP}$ with QP of 22, (b) $group_{NCP}$ with QP of 37, (c) $group_{CP}$ with QP of 22, and (d) $group_{CP}$ with QP of 37.

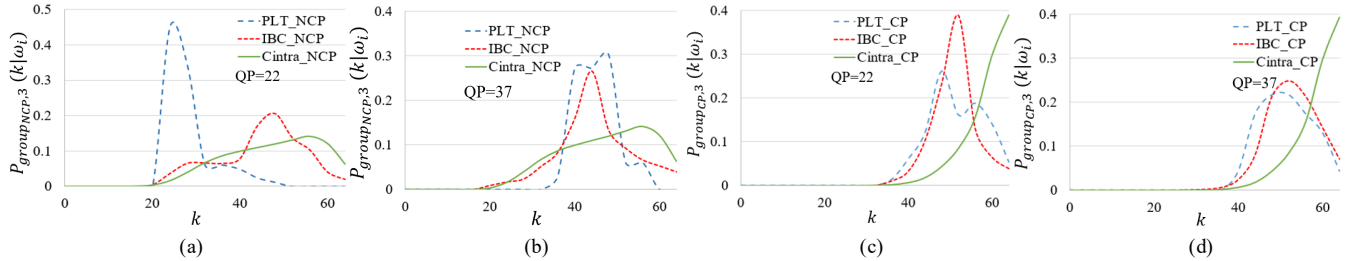


Fig. 7. Likelihood functions $P_{group_m,d}(k|\omega_i)$ for "EBURainFruits" at the depth level of 3 in (a) $group_{NCP}$ with QP of 22, (b) $group_{NCP}$ with QP of 37, (c) $group_{CP}$ with QP of 22, and (d) $group_{CP}$ with QP of 37.

large value of k. In Fig. 7, the CUs selecting Cintra mode also have larger values of k in both groups. Besides, since almost all CUs are NIBs, the curves for IBC and PLT modes are given based on very limited CUs. Cintra mode will dominate the mode decision process for "EBURainFruits", which will be shown in Fig. 10.

In considering whether to perform early skip for a mode, it is determined based on the posteriori probability $P_{group_m,d}(\omega_i|k)$, which is the conditional probability that ω_i is the best mode given k in $group_m$ and d. According to the Bayes' rule, $P_{group_m,d}(\omega_i|k)$ can be calculated as

$$P_{group_m,d}(\omega_i|k) = \frac{P_{group_m,d}(k|\omega_i)P_{group_m,d}(\omega_i)}{P_{group_m,d}(k)} \quad (4)$$

where $P_{group_m,d}(\omega_i)$ represents the priori probability of ω_i and $P_{group_m,d}(k)$ represents the total probability density of k in $group_m$ and d. $P_{group_m,d}(k)$ is obtained by

$$P_{group_m,d}(k) = \sum_{\omega_i} P_{group_m,d}(k|\omega_i)P_{group_m,d}(\omega_i). \quad (5)$$

$P_{group_m,d}(k|\omega_i)$ and $P_{group_m,d}(\omega_i)$ can be obtained from the encoding statistics in the learning frames.

Figs. 8-10 show $P_{group_m,d}(\omega_i|k)$ in $group_{NCP}$ and $group_{CP}$ for "MissionControlClip3", "Programming" and "EBURainFruits" with QPs of 22 and 37 in the depth level of 3, respectively. In $group_{CP}$ of Figs. 8-9(c) and (d) in "MissionControlClip3" and "Programming", because PLT and IBC modes are specially designed for SCBs, $P_{group_{CP},3}(\omega_{PLT}|k)$ and $P_{group_{CP},3}(\omega_{IBC}|k)$ are larger for small values of k, while the posteriori probability of Cintra mode $P_{group_{CP},3}(\omega_{Cintra}|k)$ increases as k gets larger. In $group_{NCP}$ of Figs. 8-9(a) and (b) in "MissionControlClip3" and "Programming", there are many CUs with relatively smooth content, so that $P_{group_{NCP},3}(\omega_{PLT}|k)$ are small for all k values. While smooth CUs with a small value of k can be encoded efficiently by both IBC and Cintra modes, $P_{group_m,d}(\omega_{IBC}|k)$ decreases as k gets larger, and it makes ω_{Cintra} become the most probable class for CUs with large values of k. From the above analysis, it can be found that posteriori probabilities of ω_i in $group_{NCP}$ and $group_{CP}$ do not share the same distribution, but they have a very similar distribution across the two screen content sequences, "MissionControlClip3" and "Programming", at different QPs . It implies that CP is a good feature to characterize scene contents, and the use of CP to divide CUs into $group_{NCP}$ and $group_{CP}$ is crucial to apply the Bayesian decision rule for mode decision in SCC properly. It is noted that, in "EBURainFruits" shown in Fig. 10, the most probable class is always Cintra in both $group_{NCP}$ and $group_{CP}$ since the priori probability of IBC and PLT modes are very small. Although different types of sequences have different characteristics, the proposed online-learning-based algorithm

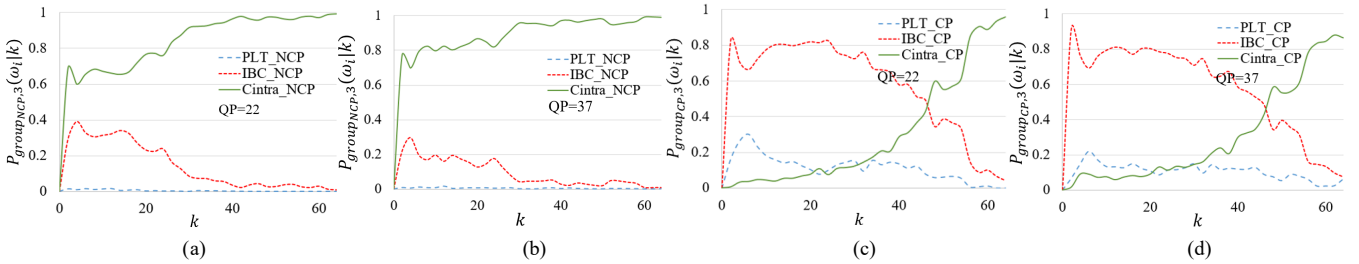


Fig. 8. Posteriori probabilities $P_{group_m,d}(\omega_i|k)$ in $group_{NCP}$ with QP of 22 (a) and 37 (b), in $group_{CP}$ with QP of 22 (c) and 37 (d) for "MissionControlClip3" at the depth level of 3.

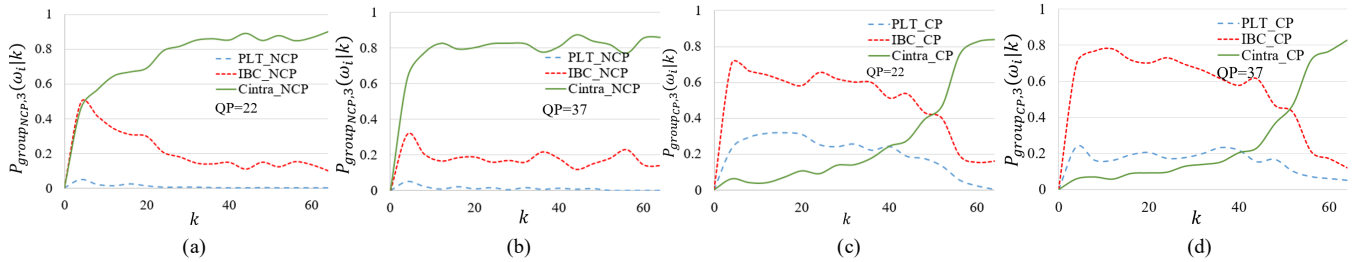


Fig. 9. Posteriori probabilities $P_{group_m,d}(\omega_i|k)$ in $group_{NCP}$ with QP of 22 (a) and 37 (b), in $group_{CP}$ with QP of 22 (c) and 37 (d) for "Programming" at the depth level of 3.

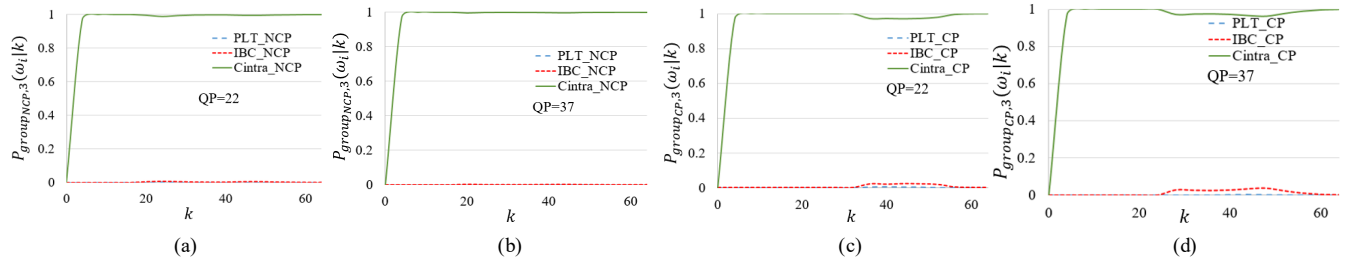


Fig. 10. Posteriori probabilities $P_{group_m,d}(\omega_i|k)$ in $group_{NCP}$ with QP of 22 (a) and 37 (b), in $group_{CP}$ with QP of 22 (c) and 37 (d) for "EBURainFruits" at the depth level of 3.

TABLE II
TEST SEQUENCES IN EACH CATEGORY

Categories	Sequences
TGM	FlyingGraphics, 1920×1080, 300 frames Desktop, 1920×1080, 600 frames Console, 1920×1080, 600 frames ChineseEditing, 1920×1080, 600 frames WebBrowsing, 1280×720, 300 frames Map, 1280×720, 600 frames Programming, 1280×720, 600 frames SlideShow, 1280×720, 500 frames
M	BasketballScreen, 2560×1440, 300 frames MissionControlClip2, 2560×1440, 300 frames MissionControlClip3, 1920×1080, 600 frames
A	Robot, 1280×720, 300 frames
CC	EBURainFruits, 1920×1080, 250 frames Kimono1, 1920×1080, 120 frames

can derive content adaptive rules for fast mode decision, which is the advantage of our algorithm compared with the pre-trained model or pre-tuned heuristics in [17-27].

When implementing the proposed FMD, $P_{group_m,d}(\omega_i|k)$ is calculated based on (4) and (5) in the learning phase. Then, to make fast mode decision, the distinct color number k for each CU is extracted, and the mode class ω_i of the CU is skipped if

$$P_{group_m,d}(\omega_i|k) < \alpha. \quad (6)$$

As illustrated in the module 4 of Fig. 2, if the probability for a CU selecting a mode class ω_i is lower than the threshold value of α, early skip for this mode will be performed. Since CUs with $k = 1$ can be encoded by all three modes efficiently, we exclude them in our algorithm for the sake of simplicity. Besides, the statistical parameters are estimated for different sizes of CUs including 64×64, 32×32, 16×16 and 8×8, respectively. The details will be given in Section III.D.

Table III
AVERAGE TIME DISTRIBUTION AMONG DIFFERENT MODES AND DEPTH LEVELS WITH QPS OF 22, 27, 32, 37

CU size	IBC (%)	PLT (%)	Cintra (%)	Total (%)
64×64	3.34		4.84	8.19
32×32	5.23	5.36	4.63	15.23
16×16	14.52	4.67	7.58	26.76
8×8	23.82	4.27	21.74	49.82

3) Mode Decision Refinement (MDR)

To further reduce the computational complexity of the mode decision process, we analyze the encoding time distribution among different modes and depth levels under AI configuration. The test sequences are the typical YUV 4:4:4 screen content sequences which were used by the experts in the JCT-VC group [29]. They are divided into 4 categories: text and graphics with motion (TGM), mixed content (M), animation (A), and camera-captured content (CC), as shown in Table II. The test platform used for simulations is a HP EliteDesk 800 G1 computer with a 64-bit Microsoft Windows 10 OS running on an Intel Core i7-4790 CPU of 3.6 GHz and 32.0 GB RAM. Table III shows the average encoding time distribution for all test sequences, where the first 100 frames of each sequences are encoded with QPs of

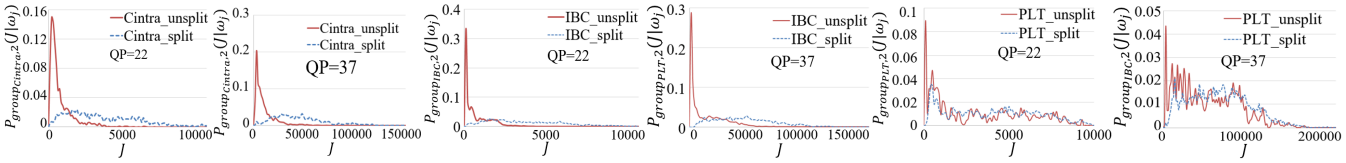


Fig. 11. Likelihood functions $P_{group_n,d}(J|\omega_j)$ for "MissionControlClip3" at depth level of 2 in (a) $group_{Cintra}$ with QP of 22, (b) $group_{Cintra}$ with QP of 37, (c) $group_{IBC}$ with QP of 22, (d) $group_{IBC}$ with QP of 37, (e) $group_{PLT}$ with QP of 22, and (f) and $group_{PLT}$ with QP of 37.

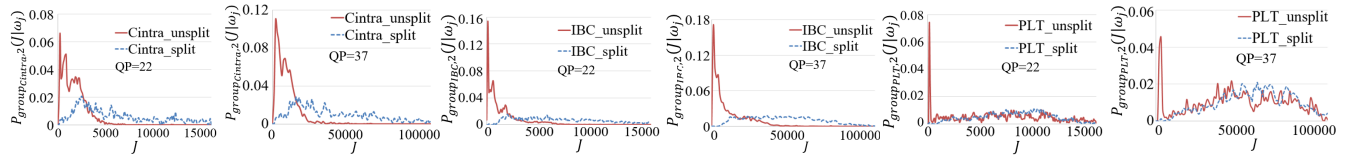


Fig. 12. Likelihood functions $P_{group_n,d}(J|\omega_j)$ for "Programming" at depth level of 2 in (a) $group_{Cintra}$ with QP of 22, (b) $group_{Cintra}$ with QP of 37, (c) $group_{IBC}$ with QP of 22, (d) $group_{IBC}$ with QP of 37, (e) $group_{PLT}$ with QP of 22, and (f) and $group_{PLT}$ with QP of 37.

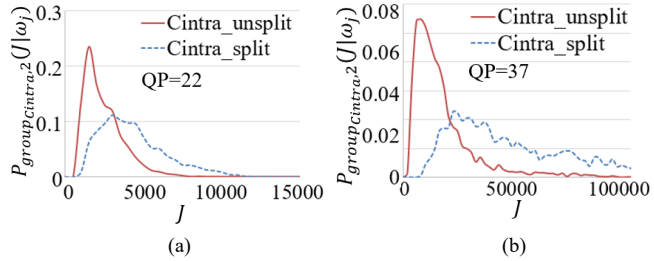


Fig. 13. Likelihood functions $P_{group_n,d}(J|\omega_j)$ in $group_{Cintra}$ with QP of (a) 22 and 37 (b) for "EBURainFruits" at depth level of 2.

Table IV
STATISTIC PROBABILITIES OF INDEPENDENT CUS FOR IBC AND CINTRA MODES WITH SIZE OF 8×8 AND 16×16.

QP	IBC8×8 (%)	Cintra8×8 (%)	IBC16×16 (%)	Cintra16×16 (%)
22	4.51	3.38	4.76	2.53
27	4.62	3.25	4.72	2.81
32	4.46	3.12	4.84	3.02
37	4.18	3.13	4.82	3.03

22, 27, 32, and 37. As shown in Table III, the computational complexity mainly comes from IBC and Cintra modes at the depth levels of 2 and 3. Therefore, a IBC and Cintra mode decision refinement algorithm for 8×8 and 16×16 CUs is worth developing.

As shown in Fig. 1, the current CU may share similar characteristics as its neighboring CUs, and it tends to select the same optimal mode of its neighboring CUs at the same depth level. If a CU selects a different mode from its top and left CUs, it is defined as the independent CU in our paper. To reveal the strong mode correlation among current CU and its top and left CUs, we study the statistic probabilities of the independent CU for IBC and Cintra modes at the depth levels of 2 and 3, which are calculated as the percentages of the independent CUs in all CUs from the depth levels of 2 and 3. The average independent CU probabilities calculated using the first 100 frames of all test sequences are shown in Table IV for four QPs. According to this table, the independent CU probabilities for IBC and Cintra modes are very low. Thus, during the mode decision refinement process, the left and top CUs at the same depth level of the current CU are used to refine the mode decision process and skip IBC and Cintra modes adaptively for 8×8 and 16×16 size CUs.

Let $mode_x$ represent a mode, and $mode_x \epsilon \{Cintra, IBC\}$. In the module 5 of Fig. 2, $mode_x$ will be skipped in the proposed algorithm if all the following conditions are satisfied:

Condition 1: Neither the top CU nor the left CU selects $mode_x$ as its optimal mode at the current depth level.

Condition 2: The mode decisions of the top and left CUs are not changed by using MDR, and that is to avoid the effect of error propagation.

Condition 3: $P_{group_m,d}(\omega_i|k) < \gamma$, $i \epsilon \{Cintra, IBC\}$. γ is a threshold used to avoid the skipping of modes with high probabilities to be the optimal mode for a CU.

B. Fast CU Size Decision by Online-learning (FCUSD)

As shown in Table III, the computational complexity increases as the depth level increases. Therefore, a CU partition early termination algorithm is desirable to skip the CU checking of the high depth level adaptively.

The CU partition process is treated as a binary classification problem in our proposed algorithm. For each CU, two classes are defined as ω_{split} and $\omega_{unsplit}$. CUs in ω_{split} will continue splitting while CUs in $\omega_{unsplit}$ will be terminated at the current depth level. The RD cost of the optimal mode under the current depth level, J, is selected as the feature for classification, and is defined as

$$J = D_{SSE} + \lambda \times R \tag{7}$$

where D_{SSE} denotes the sum of squared errors between the current CU and the predicted CU, R is the bit cost, and λ is the Lagrange multiplier in terms of QP, which is given by

$$\lambda = C \times (\frac{QP-12}{3.0})^2 \tag{8}$$

where C is a factor determined by the picture type and coding structure. In our paper, we directly extract the value of J from the original SCC encoder without inducing any further computation.

Unlike camera-capture videos which are encoded by Cintra mode only, there are different types of CUs in screen content videos, and they are encoded by different optimal modes. Considering that the distributions of RD costs can be different when CUs are encoded by three different optimal modes –

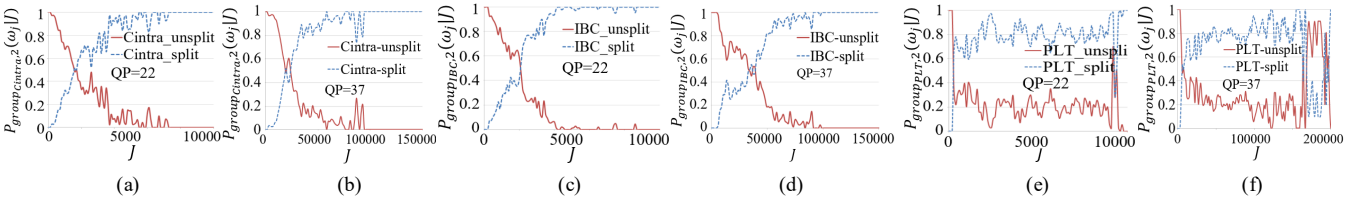


Fig. 14. Posteriori probabilities $P_{group_n,d}(\omega_j|J)$ for "MissionControlClip3" at depth level of 2 in (a) $group_{Cintra}$ with QP of 22, (b) $group_{Cintra}$ with QP of 37, (c) $group_{IBC}$ with QP of 22, (d) $group_{IBC}$ with QP of 37 (d), $group_{PLT}$ with QP of 22, and (f) $group_{PLT}$ with QP of 37.

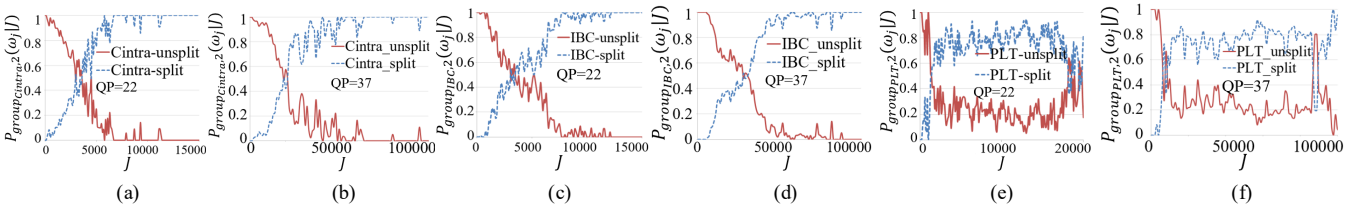


Fig. 15. Posteriori probabilities $P_{group_n,d}(\omega_j|J)$ for "Programming" at depth level of 2 in (a) $group_{Cintra}$ with QP of 22, (b) $group_{Cintra}$ with QP of 37, (c) $group_{IBC}$ with QP of 22, (d) $group_{IBC}$ with QP of 37 (d), $group_{PLT}$ with QP of 22, and (f) $group_{PLT}$ with QP of 37.

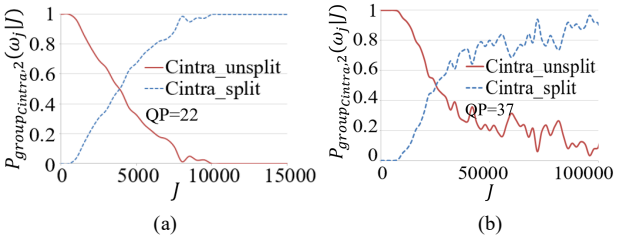


Fig. 16. Posteriori probabilities $P_{group_n,d}(\omega_j|J)$ in $group_{Cintra}$ with QP of (a) 22 and (b) 37 for "EBURainFruits" at depth level of 2.

Cintra, IBC, and PLT, CUs with the same optimal mode are then grouped together. Therefore, CUs are divided into three groups according to their optimal modes. Similar to Section III.A, likelihood functions $P_{group_n,d}(J|\omega_j)$ are estimated in the online-learning phase according to the conditional density of J in ω_j, where $j\epsilon\{split, unsplit\}$, non-SCU depth level $d\epsilon\{0,1,2\}$ and $n\epsilon\{Cintra, IBC, PLT\}$. To study the different distributions of the RD cost for CUs belonging to ω_{split} and $\omega_{unsplit}$, $P_{group_n,d}(J|\omega_j)$ for "MissionControlClip3", "Programming" and "EBURainFruits" with QPs of 22 and 37 are shown in Figs. 11-13 at the depth level of 2. Specially, only $P_{group_{Cintra},d}(J|\omega_j)$ is shown in Fig. 13 since CUs in $group_{IBC}$ and $group_{PLT}$ are very limited in "EBURainFruits". It is observed that $P_{group_n,2}(J|\omega_{split})$ in each group shows relatively wide and flat distribution, while $P_{group_n,2}(J|\omega_{unsplit})$ is concentrated in the range with small values of J. Besides, sequences encoded with QP of 37 usually have higher RD cost than QP of 22. Let $P_{group_n,d}(\omega_j)$ represent the priori probability of ω_j and $P_{group_n,d}(J)$ represent the total probability density of J, which are both obtained from the encoding statistics in the online-learning phase, then the posteriori probability $P_{group_n,d}(\omega_j|J)$ for early terminating CU partitions can be calculated by Bayes' rule as

$$P_{group_n,d}(\omega_j|J) = \frac{P_{group_n,d}(J|\omega_j)P_{group_n,d}(\omega_j)}{P_{group_n,d}(J)} \quad (9)$$

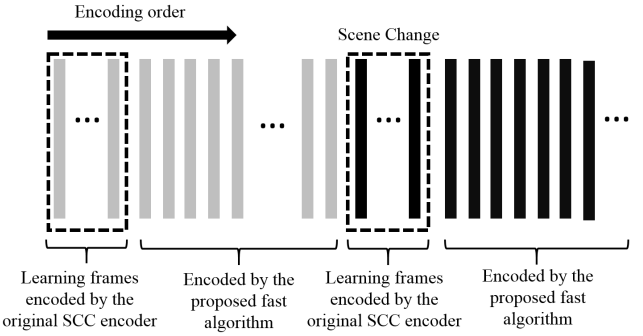


Fig. 17. Learning frame selection when encoding a video sequence (frames in the same scene are denoted by the same color).

where $P_{group_n,d}(J)$ is obtained by

$$P_{group_n,d}(J) = \Sigma_{\omega_j} P_{group_n,d}(J|\omega_j)P_{group_n,d}(\omega_j). \quad (10)$$

From (9) and (10), the posteriori probabilities $P_{group_n,d}(\omega_j|J)$ for "MissionControlClip3", "Programming" and "EBURainFruits" with QPs of 22 and 37 are shown in Figs. 14-16 at the depth level of 2. As we can see, the posteriori probability distributions are different in three groups. In each group, $P_{group_n,2}(\omega_{unsplit}|J)$ is very large for small values of J, while $P_{group_n,2}(\omega_{split}|J)$ gets larger for CUs with a higher value of J. The reason is that CUs with a small value of J are usually efficiently encoded under the current depth level, and the further partitions are unnecessary. Therefore, CUs with a small value of J are more likely to belong to $\omega_{unsplit}$. However, it is also observed that CUs with a large value of J still have relatively high probabilities to belong to $\omega_{unsplit}$ in $group_{PLT}$, as shown in Figs. 14-15(e) and (f). To achieve good prediction accuracy, FCUSD is only applied to CUs with Cintra and IBC modes in our algorithm.

After getting the statistical parameters of $P_{group_n,d}(J|\omega_j)$ and $P_{group_n,d}(\omega_j)$ in the online-learning phase, $P_{group_n,d}(\omega_j|J)$ is calculated based on (9) and (10). Meanwhile, the remaining partitions of a CU is terminated if

$$P_{group_n,d}(\omega_{split}|J) < \beta \quad (11)$$

where β is the decision threshold for early CU termination, and it determines the trade-off between the coding efficiency and computational complexity. As shown in the module 6 of Fig. 2, the RD cost, J, is extracted for the CU being encoded. We assume that the current CU is encoded efficiently, and the remaining partitions will be skipped if (11) is satisfied. The statistical parameters $P_{group_n,d}(J|\omega_j)$ and $P_{group_n,d}(\omega_j)$ are estimated for each different size of non-SCUs that are 64×64, 32×32, and 16×16 respectively in the online-learning phase. The details will be given in Section III.D.

C. Scene Change Detection for Learning Frame Updating

The Bayesian classifier is an optimal classifier based on the assumption that all statistical parameters such as $P_{group_m,d}(k|\omega_i)$, $P_{group_m,d}(\omega_i)$, $P_{group_n,d}(J|\omega_j)$ and $P_{group_n,d}(\omega_j)$ in (4) and (9) are obtained correctly. Fig. 17 shows the online-learning approach used in the proposed algorithm. For screen content videos, scene changes occur frequently such as document opening or closing, slideshow playing, etc. Then the statistical parameters will change suddenly, and it leads to wrong classifications. Therefore, scene change detection should be applied to update the statistical parameters adaptively.

To perform scene change detection in a camera-captured video sequence, two typical correlation measurement methods, difference of histogram (DOH) and histogram of difference (HOD), are introduced in [9], [30]. For DOH, it calculates the absolute sum of the histogram difference between two adjacent frames, F_a and F_b, by using luma samples, and the value of DOH is given as the ratio of the absolute sum of the histogram difference to all histograms of F_a

$$DOH(F_a, F_b) = \frac{\sum_{l=0}^{q-1} |h_a(l) - h_b(l)|}{\sum_{l=0}^{q-1} h_a(l)} \quad (12)$$

where q is the number of luma level, h_a and h_b are the histograms of F_a and F_b.

For HOD, the histogram of difference between two adjacent frames, $F_a - F_b$, is defined by $hod(l)$, where $l \in [-q+1, q-1]$. The further the histogram of luma difference is distributed from the origin of $hod(l)$, the more different the frames are. The value of HOD is given as the ratio of the sum of $hod(l)$ with l larger than a threshold τ to all histograms of $hod(l)$

$$HOD(F_a, F_b) = \frac{\sum_{l \notin [-\tau,\ \tau]} hod(l)}{\sum_{l=-q+1}^{q-1} hod(l)} \quad (13)$$

where τ is a threshold to determine the closeness to zero, and it is set to 32 in [9], [30]. If the value of DOH or HOD is larger than a threshold φ, a scene change is regarded to occur. However, for screen content videos, different scenes may have similar background, which makes these methods unable to detect scene changes. Besides, zoom and content color change situations often occur in screen content videos, but they can be considered as the same scene and the statistical parameters of the previous learning frames can still be used. If these methods are used in SCC, false scene changes might be detected.

Since frames in the same scene have similar distribution of distinct color number, a simple and efficient scene change detection method is tailor made for screen content videos in this paper. A frame is divided into non-overlapping 32×32 blocks, then the distinct color number is calculated for each block. Let F_l denotes the recent learning frames, and then the ratio of the new distinct color number (RDN) to the total distinct color number in a frame F_a is defined as

$$RDN(F_a, F_l) = \frac{\sum_{k \notin S_{F_l}^{cp}} block_{F_a}^{cp}(k) + \sum_{k \notin S_{F_l}^{ncp}} block_{F_a}^{ncp}(k)}{\sum_{k=2}^{32\times32} block_{F_a}(k)} \quad (14)$$

where $block_{F_a}^{cp}(k)$ and $block_{F_a}^{ncp}(k)$ represent the number of blocks in F_a that have k distinct colors with and without CPs, respectively, and k varies from 2 to the total pixel number in a block (32×32). Blocks with only one color are usually background, so they are excluded here. $S_{F_l}^{cp}$ and $S_{F_l}^{ncp}$ represent the distinct color number spaces with and without CPs of F_l, respectively. Thus, $\sum_{k \notin S_{F_l}^{cp}} block_{F_a}^{cp}(k)$ and $\sum_{k \notin S_{F_l}^{ncp}} block_{F_a}^{ncp}(k)$ denote the total number of the new distinct color blocks with and without CPs in F_a, respectively. $block_{F_a}(k)$ is the number of blocks in F_a that have k distinct colors, and $\sum_{k=2}^{32\times32} block_{F_a}(k)$ denotes the total distinct color number of the non-background blocks in F_a. If $RDN(F_a, F_l)$ is larger than a threshold φ, a large area in F_a has different color characteristic, and F_a is regarded as a new scene. Then, a predefined L learning frames, which is set to 2 in the proposed algorithm, are selected starting from F_a for learning and updating the statistical parameters, as shown in the modules 1 and 2 of Fig. 2. φ was empirically determined as 0.15 for typical screen content videos. However, for animation videos, they are more similar to camera-captured videos, and scene changes usually occur with a larger value of φ. Thus, a larger φ is set to 0.3 empirically for animation and camera-captured videos. It is noted that full RD optimization is performed in learning frames to get precise estimation in (4), (5), (9) and (10).

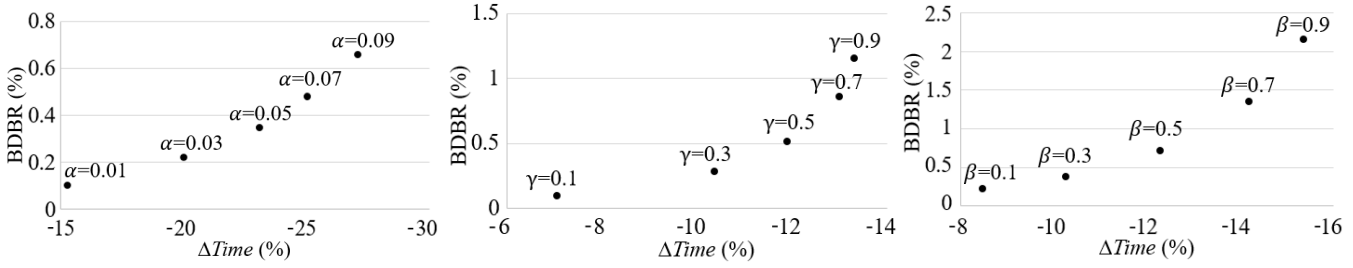
D. Likelihood Estimation and Memory Analysis

In the online-learning phase, the first L frames at the beginning of a new scene are used to estimate the statistical parameters. As shown in Figs. 5-7 and Figs. 11-13, $P_{group_m,d}(k|\omega_i)$ and $P_{group_n,d}(J|\omega_j)$ are difficult to be represented by using specific probability functions. Alternatively, the likelihood functions are estimated by a nonparametric estimation method:

$$P_{group_m,d}(k|\omega_i) = \frac{N_{group_m,d}^{\omega_i,k}}{\sum_k N_{group_m,d}^{\omega_i,k}} \quad (15)$$

$$P_{group_n,d}(J|\omega_j) = \frac{N_{group_n,d}^{\omega_j,J}}{\sum_J N_{group_n,d}^{\omega_j,J}} \quad (16)$$

where $N_{group_m,d}^{\omega_i,k}$ denotes the number of CUs with k distinct colors and belonging to the class ω_i in $group_m$ and the depth



(a) (b) (c)

Fig. 18. Average BDBR and Δ*Time* (%) of all sequences coded by individual techniques: (a) FMD, (b) MDR, and (c) FCUSD with various values of thresholds.

level of d. $N_{group_n,d}^{\omega_j,J}$ denotes the number of CUs with the RD cost of J and belonging to the class ω_j in $group_n$ and the depth level of d.

To estimate the likelihood function $P_{group_m,d}(k|\omega_i)$, numbers of CUs with k distinct colors in 3 different classes, ω_{PLT}, ω_{IBC}, and ω_{Cintra}, are recorded in the learning frames. For CUs with sizes of 8×8, 16×16 and 32×32, there are at most 64, 256 and 1024 different distinct colors, respectively, and 3 different optimal modes. For CUs with size of 64×64, there are at most 4096 different distinct colors, and 2 optimal modes (Notice that PLT mode is not applied to the CU size of 64×64). Since the number of 64×64 CU training samples is relatively small, k values of 64×64 CU size training samples are restricted to 1024 bins with 4 values in each bin to estimate the statistical fluctuation. Besides, there are 2 groups defined as regions with and without CPs. Defined as double-precision floating point which requires 8 bytes (B) in C language, 95 KB$\{[(64+256+1024)\times 3+1024\times 2]\times 2\times 8\div 1024\}$ are required to store those parameters. To estimate the likelihood function $P_{group_n,d}(J|\omega_j)$, CUs with the RD cost of J belonging to 2 different classes, $\omega_{unsplit}$ and ω_{split}, are recorded in the learning frames. Besides, there are 3 depth levels, and 2 groups defined as CUs encoded by Cintra mode and IBC mode. Since CUs belonging to $\omega_{unsplit}$ show a concentrated distribution centered in a narrow range with small RD cost, FCUSD is only applied to CUs with RD cost values smaller than 20000. For the ease of implementation, the RD cost values are restricted to 100 bins. Thus, to store these parameters, about 9.4 KB $(100\times 2\times 3\times 2\times 8\div 1024)$ are required. In total, the memory cost for likelihood function estimation is about 104.4KB, and it is acceptable to the SCC encoder.

In comparison with the existing fast SCC algorithms [23], [24], [26], they consume much less memory (less than 1 KB) than ours because they only derive several fixed rules which are independent from video content. On the other hand, the algorithms [22], [25] also need to record the information from the collocated CUs and they need more memory than ours. In [22], the depth levels, optimal modes and sample values of the last frame are required. For a video with the resolution of 2560×1440 pixels, the total memory cost is 14,850KB by using integer type which requires 4 B in C language. Specifically, the memory cost for storing sample values is 14,400KB $(2560\times 1440\times 4\div 1024)$, while the memory cost for storing depth level and the optimal mode is 450KB $(2560\times 1440\div(8\times 8)\times 2\times 4\div 1024)$ by recording them for each 8×8 block. In [25], the depth levels of the previous frame are required. Similarly, for a video with the resolution of 2560×1440 pixels, the total memory cost is 225KB $(2560\times 1440\div(8\times 8)\times 4\div 1024)$ by recording them for each 8×8 block.

E. Summary of the Proposed Algorithm

As a summary, the proposed algorithm is divided into online-learning phase and fast decision phase, as shown in Fig. 2. In the online-learning phase, the learning frames are encoded by the original SCC encoder to obtain the learning statistics. Then the learning statistics are utilized to build the Bayesian classifiers, and the fast encoding process is performed based on the classifiers in the fast decision phase. Besides, before encoding a frame in the fast decision phase, scene change detection is carried out to guarantee the strong correlation between the frames being encoded and the learning frames. To make fast encoding decisions, CP detection is firstly performed as a pre-processing step, as shown in the module 3 of Fig. 2. Then FMD, MDR, and FCUSD techniques are executed one by one to speed up the SCC encoding process, as shown in the modules 4, 5 and 6 of Fig. 2, respectively.

IV. Experimental Results and Discussion

The proposed FMD, MDR, and FCUSD techniques have been implemented in the HEVC reference software SCM-7.0 [31]. To evaluate their performances, the computational complexity reduction and Bjøntegaard delta bitrate (BDBR) [32] with QPs at 22, 27, 32, and 37 were compared with that of the original SCM-7.0, and some recent algorithms in [22-26]. The testing condition was based on AI configuration and strictly follows the Common Test Conditions (CTC) defined in [28]. The encoding time reduction, Δ*Time*, is used to measure the computational complexity reduction, which is defined as

$$\Delta Time=\frac{Time_{test}-Time_{reference}}{Time_{reference}}\times 100\% \quad (17)$$

where $Time_{test}$ represents the encoding time of the tested algorithms, and $Time_{reference}$ represents that of SCM-7.0. It is noted that negative value of $\Delta Time(\%)$ denotes encoding time decrement in percentage compared with SCM-7.0.

A. Results of Proposed Individual Techniques

From Section III, it is seen the proposed algorithm is computationally scalable, and users can select the values of α, β and γ based on their purposes. Fig. 18 shows the BDBR against Δ*Time* for different values of thresholds. It is observed that for each individual technique, larger encoding time reduction is provided as the value of the corresponding threshold

Table V
ΔTime and BDBR OF DIFFERENT ALGORITHMS COMPARED WITH SCM-7.0 FOR YUV 4:4:4 SEQUENCES

Sequence	Zhang [22]		Zhang [23]		Duanmu [24]		Lei [25]		Yang [26]		OURS1		OURS2	
	BDBR (%)	ΔTime (%)	BDBR (%)	ΔTime (%)	BDBR (%)	ΔTime (%)	BDBR (%)	ΔTime (%)	BDBR (%)	ΔTime (%)	BDBR (%)	ΔTime (%)	BDBR (%)	ΔTime (%)
FlyingGraphics	0.84	-5.03	0.54	-4.02	0.98	-19.93	1.72	-17.79	5.32	-31.32	0.19	-16.35	1.03	-25.18
Desktop	1.93	-46.82	0.67	-5.90	2.20	-26.33	1.97	-23.64	6.08	-36.05	0.55	-23.78	1.25	-35.94
Console	3.37	-38.09	2.64	-8.01	1.87	-27.82	2.87	-23.07	7.38	-42.92	0.20	-26.98	0.75	-39.49
ChineseEditing	0.64	-49.01	0.14	-5.02	1.11	-17.49	0.99	-19.08	4.30	-34.27	0.17	-23.23	0.73	-37.94
WebBrowsing	1.52	-50.67	0.27	-7.48	1.41	-28.04	5.48	-26.80	5.00	-53.55	-0.08	-28.55	0.67	-41.11
Map	0.85	-36.22	0.96	-11.46	1.56	-18.98	1.23	-19.80	2.84	-41.89	0.22	-20.18	1.29	-33.51
Programming	1.15	-39.12	0.44	-19.21	1.89	-22.08	2.50	-22.37	4.71	-27.56	0.13	-12.45	0.49	-17.79
SlideShow	1.39	-43.61	0.36	-47.24	2.88	-52.76	2.32	-55.32	3.69	-34.05	1.43	-27.57	2.33	-31.77
BasketballScreen	1.08	-41.32	0.45	-13.15	1.25	-22.57	1.46	-24.30	3.00	-31.28	0.21	-20.78	1.47	-34.12
MissionControlClip2	1.27	-38.70	0.40	-21.43	2.86	-34.33	1.71	-33.5	2.51	-38.85	0.51	-21.7	1.69	-32.54
MissionControlClip3	1.03	-38.97	0.37	-12.13	2.05	-24.71	1.69	-24.86	2.90	-34.37	0.42	-22.02	1.26	-32.74
Robot	0.93	-12.09	0.43	-18.77	1.18	-30.13	5.21	-46.90	0.59	-28.29	0.65	-29.35	1.61	-46.42
EBURainFruits	0.71	-16.76	0.21	-18.89	0.88	-27.42	1.76	-48.36	0.17	-25.95	0.17	-42.87	0.35	-58.74
Kimono1	0.14	-1.01	0.14	-26.35	1.23	-26.93	1.52	-75.49	0.13	-36.27	0.10	-40.73	0.15	-46.31
Average (TGM+M)	1.37	-38.87	0.66	-14.10	1.82	-26.82	2.18	-26.41	4.34	-36.92	0.36	-22.14	1.18	-32.92
Average (A+CC)	0.59	-9.95	0.26	-21.34	1.10	-28.16	2.83	-56.92	0.30	-30.17	0.37	-37.65	0.70	-50.49
Average (ALL)	1.20	-32.67	0.57	-15.65	1.67	-27.11	2.32	-32.95	3.47	-35.47	0.35	-25.47	1.08	-36.69

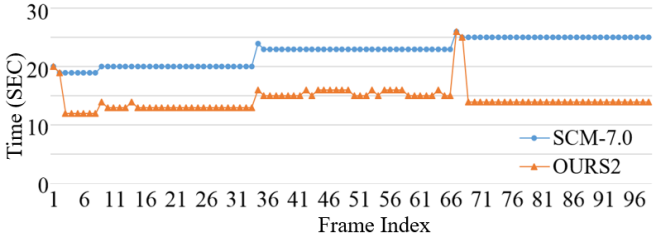


Fig. 19. Encoding time of each frame for "WebBrowsing" with QP of 22.

increases. However, it also brings a higher increase in BDBR. Specifically, FMD provides the largest encoding time reduction among the three techniques with negligible coding efficiency loss. When α varies from 0.01 to 0.09, FMD provides 15.27% to 27.06% encoding time reduction with 0.10% to 0.66% increase in BDBR. When γ varies from 0.1 to 0.9, MDR provides 7.05% to 13.31% encoding time reduction with 0.10% to 1.16% increase in BDBR. When β varies from 0.1 to 0.9, FCUSD provides 8.48% to 15.35% encoding time reduction with 0.22% to 2.16% increase in BDBR.

B. Results of Overall Algorithm

In this section, the performance of the proposed overall algorithm is given in Table V. To have a good tradeoff between computational complexity and RD performance, it is suggested that the values of α, β and γ are selected to let each technique has similar increase in BDBR. In our paper, two settings are tested, α=0.01, γ=0.1, β=0.1 and α=0.07, γ=0.3, β=0.3. They are denoted as OURS1 and OURS2, respectively, for simplicity. The increases in BDBR of the three individual techniques are all smaller than 0.3% in OURS1, while the increases in BDBR are all smaller than 0.5% in OURS2. As test sequences in TGM and M contain both NIBs and SCBs while sequences in A and CC contain only NIBs, the average results for TGM+M and A+CC sequences are also provided in Table V. It is observed from Table V that OURS1 provides 25.47% encoding time reduction with 0.35% increase in BDBR while OURS2 provides 36.69% encoding time reduction with 1.08% increase in BDBR.

To further investigate the encoding time reduction performance of the proposed algorithm, the encoding time of OURS2 for "WebBrowsing" was compared with the original SCM-7.0 frame by frame, and the results are shown in Fig. 19. In the online-learning phase, the learning frames are encoded by the original intra prediction process to obtain the learning statistics. Therefore, it can be observed that the encoding time is almost the same as SCM-7.0 for the learning frames. Then, the proposed algorithm is applied to the following frames, and encoding time is reduced dramatically.

C. Performance Comparison with State-of-the-art Algorithms

This section will compare our proposed algorithm with other recent SCC algorithms. They include the algorithms in Zhang [22], Zhang [23], Duanmu [24] ("RD-preserving" setting), Lei [25], and Yang [26]. It is noted that they were implemented in different reference software from ours in their original publications. Zhang [23] was simulated using HM-12.1+RExt-5.1 rather than SCM, while Zhang [22], Duanmu [24], Lei [25] and Yang [26] were simulated using SCM-3.0, SCM-4.0 and SCM-2.0, SCM-5.0 respectively. There are numerous enhancements, speed-up techniques and codes clean-up in SCM-7.0 compared with the older versions. In the older versions, the BV signal in IBC mode was not unified with the inter mode which only has left and above BVs as predictors with no skip and merge modes. Consequently, incoming CUs always need to check the time-consuming IBC search and PLT modes without early termination. Moreover, N×N IBC search was done after 2N×N search while it is eliminated in SCM-7.0. In addition, the older versions enable PLT mode in the depth level of 0 while it is disabled in SCM-7.0 because of the occasional use. Due to those differences, we re-implemented them into SCM-7.0 for fair comparisons.

By employing online-learning-based Bayesian decision rule that updates the decision models adaptively according to the content being encoded, the proposed algorithm always shows higher encoding time reduction and smaller increase in BDBR on average than the pre-trained models and pre-tuned heuristics in [22-26]. Specially, OURS2 provides over 50% encoding time reduction with only 0.70% increase in BDBR for sequences in A+CC, and it significantly outperforms the pre-trained models and pre-tuned heuristics in [22-26]. Since learning frames only contain NIBs, the derived Bayesian classifiers can easily skip both IBC and PLT modes based on the statistics of learning frames. On the contrary, the pre-trained models and pre-tuned

Table VI
PERFORMANCE OF THE PROPOSED ALGORITHM COMBINE WITH ZHANG [22]

Sequences	OUR2		OURS2+Zhang [22]	
	BDBR (%)	ΔTime (%)	BDBR (%)	ΔTime (%)
FlyingGraphics	1.03	-25.18	1.56	-28.36
Desktop	1.25	-35.94	2.23	-62.92
Console	0.75	-39.49	2.05	-58.97
ChineseEditing	0.73	-37.94	1.67	-63.66
WebBrowsing	0.67	-41.11	2.64	-64.00
Map	1.29	-33.51	1.89	-46.62
Programming	0.49	-17.79	1.08	-29.53
SlideShow	2.33	-31.77	1.49	-34.37
BasketballScreen	1.47	-34.12	2.70	-53.14
MissionControlClip2	1.69	-32.54	2.15	-46.38
MissionControlClip3	1.26	-32.74	2.12	-52.50
Robot	1.61	-46.42	2.46	-46.71
EBURainFruits	0.35	-58.74	0.68	-51.71
Kimono1	0.15	-46.31	0.23	-44.71
Average (TGM+M)	1.18	-32.92	1.96	-49.13
Average (A+CC)	0.70	-50.49	1.12	-47.71
Average (ALL)	1.08	-36.69	1.78	-48.83

heuristics are generated using a mixed training data of NIBs and SCBs. Therefore, they fail to have such accurate prediction for sequences in A+CC as OURS2, and it leads to less coding gain than the proposed algorithm, as observed in Table V.

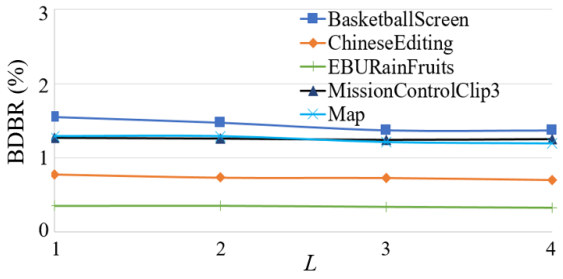
From Table V, OURS1 shows better performance on average than Zhang [23], which only has 15.65% encoding time reduction with 0.57% increase in BDBR, while OURS1 shows 25.47% encoding time reduction with 0.34% increase in BDBR. Furthermore, OURS2 shows better performance on average than Zhang [22]. In Table V, Zhang [22] provides 32.67% encoding time reduction with 1.20% increase in BDBR. The complexity savings in Zhang [22] heavily relies on the stationary CUs between adjacent frames by re-using the CU depth information from the collocated CU in the previous frame, and it induces more memory cost as mentioned in Section III.C. As a result, it works very well for sequences in TGM and M which contain many stationary regions, where 38.87% encoding time are reduced, but larger BDBR increase as compared with OURS2 is induced. It is also observed that it provides only 5.03% encoding time reduction for "FlyingGraphics". It is because "FlyingGraphics" contains rapid windows rotation and color changing. In this situation, the number of stationary CUs decreases. Due to the same reason, it provides only 9.95% encoding time reduction for sequences in A and CC. On the contrary, our proposed algorithm works well for CUs with object movements or non-stationary CUs since OURS2 does not use the collocated information from the previous frame. Its performance is then insensitive to the motion activity in the sequence. Therefore, the average time reduction of OURS2 provides more consistent savings for sequences in TGM and M, and has better performance for sequences in A and CC. When the access to the information from the collocated CU is feasible, our proposed algorithm can be integrated with the CU depth prediction in Zhang [22] such that the temporal CU depth correlation can be utilized for stationary CUs. As observed from Table VI, with the integration of Zhang [22] into OURS2, i.e. OURS2 + Zhang [22], it performs well for all types of sequences. On average, 48.83% encoding time is reduced with 1.78% increase in BBDR. This shows that Zhang [22] can speed up stationary CUs while OURS2 can work well for CUs with object movements. Further encoding time reduction achieved by the integration proves that the areas addressed by OURS2 are different from Zhang [22]. This is an excellent complement between OURS2 and Zhang [22]. From Table VI, it can be seen that the BDBR is increased when OURS2 is incorporated with Zhang [22]. It is expected as no fine-tuning for the algorithms is performed in the code merging process, which causes the fast techniques to become greedy. This integration could be a point for our immediate future work.

Table VII
COMPARITION OF THE MODE DECISION DISTRIBUTION DECIDED BY DIFFERENT ALGORITHMS WITHOUT CU SIZE DECSION

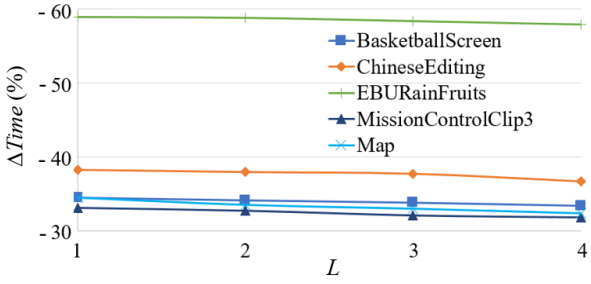
Algorithm	Depth level	ChineseEditing						
		Cintra only	IBC only	PLT only	Cintra+ IBC	Cintra + PLT	IBC+ PLT	Cintra+ IBC+PLT
OURS2	0	65.19	1.39		33.42			
	1	2.88	0.52	53.66	1.74	5.46	32.00	3.74
	2	0.93	0.33	20.55	10.34	3.35	58.98	5.52
	3	1.01	0.79	2.81	16.96	0.51	61.86	16.06
Duanmu [24]	0	100	0		0			
	1	3.28	0	0	0	0	50.36	46.36
	2	6.80	0.01	0	0.11	0	45.13	47.95
	3	0	0.14	0	6.24	0	32.44	61.18
Lei [25]	0	0	0		100			
	1	0	0	0	0	0	0	100
	2	0	0	0	0	0	0	100
	3	0.02	0	0	0	0	0	99.98
Yang [26]	0	100	0		0			
	1	69.09	0	0	0	0	0	30.90
	2	22.57	0	0	0	0	0	77.43
	3	17.99	0	0	0	0	48.97	33.04

Similar to our proposed algorithm, Duanmu [24], Lei [25] and Yang [26] also consider the whole encoding process of SCC to speed up both of the mode decision and CU size decision by classifying CUs into NIBs and SCBs. From Table V, Duanmu [24], Lei [25] and Yang [26] show 27.11%, 32.95%, 35.47% encoding time reduction with 1.67%, 2.32% and 3.57% increase in BDBR. Their approaches are more focused on the fast encoding of NIBs by checking Cintra mode only for NIBs while checking IBC+PLT or Cintra+IBC+PLT for SCBs. Comparatively, SCBs may check one mode by using the proposed algorithm. As a result, OURS2 provides 1.22%-9.57% larger encoding time reduction with 0.59%-2.39% less BDBR increment on average than their approaches. It is noted that Lei [25] also uses depth prediction based on the neighboring and collocated CUs. When OURS2 also adopts the depth prediction in [22], as mentioned above, the results in Table VI show further overwhelming performance in comparison with Lei [25] in Table V.

To further illustrate the advantage of our proposed mode decision, the mode distribution by OURS2, Duanmu [24], Lei [25] and Yang [26] are shown in Table VII. It is noted that the CU size decisions for all algorithms are disabled to clearly illustrate the mode distribution. Table VII shows the mode decision distribution of "ChineseEditing" which only contains SCBs. Results in Table VII show that OURS2 provides flexible mode decision, where a mode can be checked alone or with another mode. For example, PLT mode can be checked alone or with Cintra/IBC mode for a CU. For "ChineseEditing", 53.66% and 20.55% CUs only check PLT mode in depth levels of 1 and 2, respectively. Since the mode decision becomes more complicated for small CUs, various combinations of modes are usually checked in depth level of 3. Besides, in the depth level of 0 in which only IBC and Cintra modes are enabled, 65.19% CUs are decided to check Cintra mode only since it is difficult



(a)



(b)

Fig. 20. Performance of the proposed algorithm with the values of L of 1-4.

Table VIII
Δ*Time* and BDBR of Different Algorithms Compared With SCM-7.0 for RGB 4:4:4 and YUV 4:2:0 Sequences

Format		TGM+M		A+CC		Average	
		BDBR (%)	Δ*Time* (%)	BDBR (%)	Δ*Time* (%)	BDBR (%)	Δ*Time* (%)
RGB 4:4:4	[22]	1.02	-38.29	0.31	-9.39	0.87	-32.10
	[23]	0.51	-11.22	0.09	-12.65	0.42	-11.53
	[24]	2.19	-24.23	0.51	-26.41	1.83	-24.70
	[25]	2.18	-24.68	2.62	-55.72	2.27	-31.33
	[26]	3.17	-32.03	0.25	-22.36	2.54	-29.96
	OURS1	0.23	-20.61	0.19	-39.49	0.22	-24.66
	OURS2	0.89	-29.54	0.47	-50.36	0.80	-34.01
YUV 4:2:0	[22]	1.47	-29.36	0.97	-12.58	1.39	-26.78
	[23]	0.79	-12.77	0.27	-17.58	0.71	-13.51
	[24]	2.39	-19.80	1.92	-25.53	2.32	-20.68
	[25]	1.88	-24.31	3.22	-39.70	2.08	-26.68
	[26]	4.11	-30.65	1.36	-28.11	3.69	-30.26
	OURS1	0.29	-17.51	0.43	-32.08	0.31	-19.75
	OURS2	1.12	-25.56	1.61	-51.78	1.19	-30.44

to find repeated patterns for large CUs by IBC mode. On the other hand, the fast mode decision in Duanmu [24], Lei [25] and Yang [26] treat the decisions for IBC and PLT modes the same, and then at least two modes (IBC+PLT or Cintra+IBC+PLT) are checked for a SCB. In spite of eliminating unnecessary modes based on coding bits in [24], it only works for limited number of CUs by checking only one single mode. Due to the above arrangement, Duanmu [24] checks either both IBC and PLT modes or all modes for over 93% CUs in the depth levels of 1-3 in "ChineseEditing", as shown in Table VII. In contrast, the proposed fast mode decision technique makes decision for each mode separately, so that many SCBs can only check one mode. In this table, Duanmu [24] always checks Cintra mode in the depth level of 0 as it disables IBC mode in the depth level of 0. Lei [25] needs to check all modes for over 99% CUs in "ChineseEditing". Yang [26] needs to check Cintra mode for all CUs with 2N×2N PUs to extract features although Cintra mode can be skipped for most CUs in "ChineseEditing". Therefore, our proposed mode decision shows much better performance than their mode decision methods.

D. Results of RGB 4:4:4 and YUV4:2:0

Since we adopt an online-learning-based approach whose learning statistics are updated according to the current video content adaptively, it can be easily applied to sequences in other formats. Table VIII shows the performance comparison of sequences in RGB 4:4:4 and YUV 4:2:0 formats. Similar trend is observed in YUV 4:4:4 sequences that OURS1 provides better performance than Zhang [23], while OURS2 outperforms Zhang [22], Duanmu [24], Lei [25] and Yang [26]. Specifically, OURS1 provides 24.66% and 19.75% encoding time reduction with 0.22% and 0.31% negligible increase in BDBR for sequences in RGB 4:4:4 and YUV 4:2:0, respectively. OURS2 provides 34.01% and 30.44% encoding time reduction with 0.80% and 1.19% increase in BDBR for sequences in RGB 4:4:4 and YUV 4:2:0, respectively.

E. Impact of Scene Change Detection on Proposed Algorithm

To show the effectiveness of the proposed RDN on scene change detection, the performances of OURS2 with and without scene change detection have been carried out for comparison, Besides, we compared RDN with the 2 typical scene change detection methods introduced in the Section III.C – HOD and DOH, and the threshold φ was set to 0.25 [30] for DOH and 0.2 [9] for HOD. The results are shown in Table IX, and if scene changes are detected in a sequence by using HOD, DOH or RDN, the sequence is marked as *SC*. Otherwise, it is marked as *NSC*. We can see that RDN, HOD, DOH and the case without scene change detection provide very similar performance for *NSC* sequences, where about 43% encoding time is saved with 0.93% increase in BDBR. However, the proposed RDN can efficiently reduce the increase in BDBR for *SC* sequences, where 31.57% encoding time is saved with 1.19% increase in BDBR. Compared with HOD and the case without scene change detection, the proposed RDN provides 0.78% and 0.54% smaller increase in BDBR with only 3.83% and 2.58% time saving drop on average, respectively. Besides, the proposed RDN shows 0.23% smaller increase in BDBR with 0.48% larger encoding time reduction than DOH. The reason is that screen content videos contain many background blocks, and the conventional scene change detection methods HOD and DOH might treat two different scenes as one if they contain similar background. Besides, due to the frequent occurrences of local motion and content color change situations, false scene changes are detected in "Programming" by using HOD and in "SlideShow" by both HOD and DOH, which makes the BDBR increases even higher than that of the case without scene change detection. Furthermore, it is also observed that many false scene changes are detected in "FlyingGraphics" by DOH, which contains many color change situations, and only 3.62% time saving is provided. For the proposed RDN, the reason for the slight drop on time saving is that more frames are selected as learning frames, and FMD, MDR, and FCUSD cannot be applied to the learning frames. As a result, less computational complexity reduction can be achieved but better BDBR can be provided due to the precise statistical parameters are obtained.

To understand the impact of the number of learning frames L, Fig. 20 shows the performance of OURS2 with the values of L

Table IX
PERFORMANCE COMPARISON USING DIFFERENT SCENE CHANGE DETECTION METHODS

Sequences	Without scene change detection		HOD		DOH		Proposed RDN	
	BDBR (%)	Δ*Time* (%)	BDBR (%)	Δ*Time* (%)	BDBR (%)	Δ*Time* (%)	BDBR (%)	Δ*Time* (%)
FlyingGraphics(*SC*)	1.03	-25.20	0.95	-23.13	0.12	-3.62	1.03	-25.18
Desktop(*NSC*)	1.25	-35.77	1.25	-35.72	1.25	-35.08	1.25	-35.94
Console(*NSC*)	0.75	-39.56	0.75	-39.32	0.75	-39.22	0.75	-39.49
ChineseEditing(*SC*)	0.73	-37.58	0.73	-37.56	0.68	-33.78	0.73	-37.94
WebBrowsing(SC*)*	1.25	-40.26	0.62	-40.73	0.67	-40.56	0.67	-41.11
Map(*SC*)	3.18	-35.87	3.18	-34.84	1.53	-34.05	1.29	-33.51
Programming(*SC*)	1.67	-27.04	1.92	-26.50	1.17	-22.14	0.49	-17.79
SlideShow(*SC*)	2.90	-50.60	3.63	-48.64	3.12	-48.78	2.33	-31.77
BasketballScreen(*NSC*)	1.47	-34.22	1.47	-34.58	1.47	-34.27	1.47	-34.12
MissionControlClip2(*SC*)	2.46	-34.10	1.59	-29.29	2.46	-33.10	1.69	-32.54
MissionControlClip3(*SC*)	1.59	-32.57	1.25	-32.47	1.59	-32.67	1.26	-32.74
Robot(*NSC*)	1.61	-46.22	1.61	-46.32	1.61	-46.27	1.61	-46.42
EBURainFruits(*NSC*)	0.35	-58.72	0.35	-58.60	0.35	-58.67	0.35	-58.74
Kimono1(*NSC*)	0.15	-46.38	0.15	-46.32	0.15	-46.27	0.15	-46.31
Average(*NSC*)	0.93	-43.48	0.93	-43.48	0.93	-43.30	0.93	-43.50
Average(*SC*)	1.97	-35.40	1.73	-34.15	1.42	-31.09	1.19	-31.57

Table X
PERFORMANCE OF OURS2 WITHOUT CP DETECTION

Sequences	OUR2 without CP detection		OUR2	
	BDBR (%)	Δ*Time* (%)	BDBR (%)	Δ*Time* (%)
FlyingGraphics	0.45	-21.34	1.03	-25.18
Desktop	0.72	-25.20	1.25	-35.94
Console	0.39	-29.22	0.75	-39.49
ChineseEditing	0.53	-24.13	0.73	-37.94
WebBrowsing	0.25	-34.49	0.67	-41.11
Map	1.45	-29.85	1.29	-33.51
Programming	0.65	-17.27	0.49	-17.79
SlideShow	2.64	-31.01	2.33	-31.77
BasketballScreen	1.32	-24.33	1.47	-34.12
MissionControlClip2	2.46	-29.79	1.69	-32.54
MissionControlClip3	1.17	-21.07	1.26	-32.74
Robot	1.74	-45.95	1.61	-46.42
EBURainFruits	0.36	-58.97	0.35	-58.74
Kimono1	0.12	-46.29	0.15	-46.31
Average (TGM+M)	1.09	-26.15	1.18	-32.92
Average (A+CC)	0.74	-50.40	0.70	-50.49
Average (ALL)	1.02	-31.35	1.08	-36.69

from 1 to 4 for 5 typical sequences, and similar results are observed for other sequences. It is observed that the value of L has minor impact on the performance of proposed algorithm. Basically, the increase in BDBR is reduced as L gets larger. However, the encoding time saving is also reduced because the proposed fast algorithm is not applied to the learning frames. To balance the RD performance and time saving, L is set to 2 in this paper.

F. Impact of CP Detection on Proposed Algorithm

To characterize screen contents, CP is utilized as a good feature for the proposed algorithm. To reveal its impact, the results of OURS2 without the use of CPs are shown in Table X. It is observed that OURS2 without CP detection provides 5.34% less encoding time reduction than OURS2 with similar increase in BDBR on average. Specifically, since sequences in A+CC only contain NIBs, and Cintra mode dominates the mode decision process, CPs have very minor impact for them. However, CPs are important for sequences in TGM+M which show mixed content of NIBs and SCBs. It is observed that CPs can help to further reduce encoding time by 6.77% for sequences in TGM+M. Therefore, the result in Table X demonstrates that better performance can be achieved for screen content sequences by adopting CPs in the Bayesian decision rule for SCC.

V. Conclusions

In this paper, a fast mode and CU size decision algorithm based on online-learning using the Bayesian decision rule has been proposed to reduce the computational complexity of SCC. A new scene change detection method specially designed for screen content videos is applied to estimate the statistical parameters correctly for different scenes. Then, the proposed algorithm is applied after the learning phase, which includes three steps: FMD, MDR and FCUSD. For FMD, a CP detection method is employed to classify a frame into NIBs and SCBs roughly. Then, distinct color number in a CU is extracted as the unique feature for mode classification. In MDR, the spatial optimal mode correlation is utilized to further eliminate unnecessary mode candidates. In FCUSD, the RD cost of the current CU is used as the feature to early terminate the CU partition process. Compared with SCM-7.0, the proposed algorithm achieves 36.69% encoding time reduction with a negligible BDBR increment of 1.08% on average for typical screen content videos under AI configurations. In this work, cascaded empirical thresholds are used for multiple decisions. To simplify the framework of the proposed algorithm, future works may include deep learning-based fast SCC encoding algorithms that integrate fast mode decision and partition decision into a single model. To reduce the testing time of deep learning-based approaches, a multitask-based network, which gives both mode and CU partitioning decisions for an entire CTU in a single test, can be considered. Since this paper directly makes mode decision rather than the conventional CU type classification, it can be considered as the baseline for more advanced mode decision algorithms in the future.

REFERENCES

[1] J. Xu, R. Joshi, and R. A. Cohen, "Overview of the emerging HEVC screen content coding extension," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 50–62, Jan. 2016.

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

[3] X. Xu *et al.*, "Intra block copy in HEVC screen content coding extensions", *IEEE J. Emerg. Sel. Topic Circuits Syst.*, vol. 6, no. 4, pp. 409–419, Dec. 2016.

[4] X. Xiu, Y. He, R. Joshi, M. Karczewicz, P. Onno, C. Gisquet, and G. Laroche, "Palette-based coding in the screen content coding extension

of the HEVC standard," in *Proc. Data Compression Conf*, Snowbird, UT, USA, Apr. 2015, pp 253 – 262.

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

[6] J. Vanne, M. Viitanen, T. D. Hämäläinen, and A. Hallapuro, "Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs," *IEEE Trans. Circuits Syst. Video Technol*., vol. 22, no. 12, pp. 1885–1898, Dec. 2012.

[7] Y. Wang, X. Fan, L. Zhao, S. Ma, D. Zhao, W. Gao. "A Fast Intra Coding Algorithm for HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Paris, France, Oct. 2014, pp.4117-4121.

[8] L. Shen, Z. Zhang, and Z. Liu, "Effective CU size decision for HEVC intra coding," *IEEE Trans. Image Process.*, vol.23, no.10, pp.4232–4241, Jul. 2014.

[9] H.-S. Kim, and R.-H. Park, "Fast CU partitioning algorithm for HEVC using an online-learning-based bayesian decision rule," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 130–138 Jan. 2016.

[10] X. Shen, L. Yu, and J. Chen, "Fast coding unit size selection for HEVC based on Bayesian decision rule," in *Proc. Picture Coding Symp. (PCS)*, Krakow, Poland, May. 2012, pp. 453–456.

[11] K. Lim, J. Lee, S.Kim and S.Lee, "Fast PU skip and split termination algorithm for HEVC intra prediction." *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 8, pp. 1335 - 1346 Aug. 2015.

[12] Y. Zhang, S. Kwong, X. Wang, H. Yuan, Z. Pan and L. Xu, "Machine learning-based coding unit depth decisions for flexible complexity allocation in High Efficiency Video Coding," *IEEE Trans. Image Process*., vol.24, no.7, pp.2225-2238, Jul. 2015.

[13] B. Du, W.-C. Siu and X. Yang, "Fast CU partition strategy for HEVC intra-frame coding using learning approach via random forests," in *Proc. APSIPA ASC*, Hong Kong, Dec. 2015, pp. 1085 – 1090.

[14] H-B Zhang, Y-L Chan, C-H Fu, S-H Tsang, and W-C Siu, "Quadtree decision for depth intra coding in 3D-HEVC by good feature," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Shanghai, China, Mar. 2016. pp. 1409–1413.

[15] M. Zhang, C. Zhao, and J. Xu, "An adaptive fast intra mode decision in HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Orlando, FL, USA, Sep. 2012, pp. 221–224.

[16] A. S. Motra, A. Gupta, M. Shukla, and P. Bansal: "Fast intra mode decision for HEVC video encoder," in *Int. Conf. on Software, Telecomm. Comput. Netw. (SoftCOM)*, Split, Croatia, Sep. 2012, pp. 1–5.

[17] D.-K. Kwon, and M. Budagavi, "Fast intra block copy (IntraBC) search for HEVC screen content coding," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Melbourne VIC, Australia, Jun. 2014, pp. 9–12.

[18] S.-H. Tsang, Y.-L. Chan, and W.-C. Siu, "Hash based fast local search for intra block copy (IntraBC) mode in HEVC screen content coding," in *Proc. APSIPA ASC*, Hong Kong, Dec. 2015, pp. 396–400.

[19] S.-H. Tsang, W. Kuang, Y.-L. Chan, and W.-C. Siu, "Fast HEVC screen content coding by skipping unnecessary checking of intra block copy mode based on CU activity and gradient," in *Proc. APSIPA ASC*, Jeju, Korea, Dec. 2016, pp.1–5.

[20] S.-H. Tsang, Y.-L. Chan, and W.-C. Siu, "Fast and efficient intra coding techniques for smooth Regions in screen content coding based on boundary prediction samples," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Brisbane, Australia, Apr. 2015, pp.1409–1413.

[21] F. Duanmu, Z. Ma, and Y. Wang, "Fast CU partition decision using machine learning for screen content compression," in *Proc. IEEE Int. Conf. Image Process.*, Quebec, QC, Canada, Sep. 2015, pp. 4972–4976.

[22] H. Zhang, Q. Zhou, N.-N Shi, F. Yang, X. Feng, and Z. Ma, "Fast intra mode decision and block matching for HEVC screen content compression," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Shanghai, China, Mar. 2016, pp.1377–1381.

[23] M. Zhang, Y. Guo, and H. Bai, "Fast intra partition algorithm for HEVC screen content coding," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Valletta, Malta, Dec. 2014, pp. 390–393.

[24] F. Duanmu, Z. Ma, and Y. Wang, "Fast mode and partition decision using machine learning for intra-frame coding in HEVC screen content coding extension," *IEEE J. Emerg. Sel. Topic Circuits Syst.*, vol. 6, no. 4, pp.517–531, Dec. 2016.

[25] J. Lei, D. Li, Z, Pan, Z. Sun, S. Kwong, and C. Hou, "Fast intra prediction based on content property analysis for low complexity HEVC-based screen content coding," *IEEE Trans. Broadcast*., vol. 63, no.1, pp.48–58, Mar. 2017.

[26] H. Yang, L. Shen, and P. An, "An efficient intra coding algorithm based on statistical learning for screen content coding", in *Proc. IEEE Int. Conf. Image Process.*, Beijing, China, Sep. 2017, pp. 2468–2472.

[27] C. Huang, Z. Peng, F. Chen, Q. Jiang, G. Jiang and Q. Hu, "Efficient CU and PU Decision Based on Neural Network and Gray Level Co-Occurrence Matrix for Intra Prediction of Screen Content Coding," *IEEE Access*, vol. 6, pp. 46643 - 46655, Aug. 2018.

[28] J.-B, Shi, and C. Tomasi, "Good features to track," in *Proc. IEEE Int. Conf. Comp.* Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 1994, pp. 593–600.

[29] H.-P. Yu, R. Cohen, K. Rapaka, and J. -Z Xu, "Common test conditions for screen content coding", 21th JCT-VC meeting, document JCTVC-U1015-r2, Warsaw, Poland, Jun. 2015.

[30] J. W. Lee, and B. W. Dickinson, "Temporally adaptive motion interpolation exploiting temporal masking in visual perception," *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 513–526, Sep. 1994.

[31] HM-16.8+SCM-7.0, HEVC test model version 16.8 screen content model version 7.0, [Online], available at: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.8+SCM-7.0/.

[32] G. Bjontegaard, "Calculation of average PSNR differences between rd-curves," document VCEG-M33, VCEG, Austin, Texas, USA, Mar. 2001.