Improvements of HEVC SCC Palette Mode and Intra Block Copy

Yu-Chen Sun, Shih-Ta Hsiang, Jungsun Kim, Yi-Wen Chen, Xiaozhong Xu, Weijia Zhu, and ShawMin Lei, *Fellow, IEEE*

***Abstract*— Palette mode and intra block copy are two new coding tools that have been adopted in the screen content coding extensions of high efficiency video Ccoding (HEVC SCC). Palette mode can represent color clusters for screen content efficiently. Intra block copy (IBC) facilitates block prediction within the same frame. Both tools have demonstrated promising benefits for screen content coding. In addition to HEVC SCC, this paper introduces five new coding techniques to enhance palette mode or IBC: transition copy, prediction across coding unit boundary, advanced motion vector coding, modified temporal merge candidate derivation, and new default IBC merge candidates. Experimental results show that the proposed techniques can achieve 5.4%, 5.2%, and 4.9% BD-rate savings compared with HEVC SCC for "YUV, text & graphics with motion, 1080p & 720p sequences" under all intra, random access, and low-delay B common test conditions, respectively.**

***Index Terms*— High efficiency video coding (HEVC), intra block copy, palette mode, screen content coding (SCC).**

I. Introduction

In recent years, screen content coding has been widely used for many applications, such as remote desktop, web conferencing, and cloud computing [1]. In March 2014, the Joint Collaborative Team on Video Coding (JCT-VC), formed by ISO/IEC MPEG and ITU-T VCEG, started to standardize Screen Content Coding Extensions of High Efficiency Video Coding (HEVC SCC) [2], [3]. Three major coding tools are adopted in HEVC SCC: adaptive color transform [4], palette mode [5], and intra block copy (IBC) [6]–[10]. Adaptive color transform aims to transform video sequences, especially those in RGB format, into another color space where data can be compressed more effectively. Palette mode can efficiently describe all pixels in a coding unit (CU) with few selected representative colors. Intra block copy explores the content correlation within a frame.

Compared with HEVC, HEVC SCC is able to achieve an impressive bit-rate saving [20] for screen content. However, the need for higher compression of screen content video is still strong. Since palette mode and IBC are two of the major tools in HEVC SCC, in this paper, five new coding techniques related to palette mode and IBC are presented to further enhance the coding performance of HEVC SCC. These techniques were previously proposed for development of the HEVC SCC extension. However, the JCT-VC eventually decided to unify the IBC with the existing HEVC Inter mode [10] and preferred having a low-complexity Palette mode design in order to reduce the additional implementation cost for codec extension. These proposed techniques were not adopted into the final HEVC SCC. Nevertheless, as we will see from the experimental results in Section IV, the proposed techniques can further improve the HEVC SCC in coding efficiency and be potentially useful for technological advancement in future SCC applications. The proposed techniques are summarized as follows.

New techniques for palette mode.

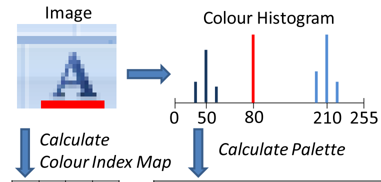
1) Transition copy (TC)—exploring the correlation between an index and all of the previously coded indexes
2) Prediction across CU boundary—utilizing the correlation between the two sides of CU boundary.

New techniques for intra block copy.

3) Advanced motion vector (MV) coding—improving the efficiency of motion vector coding.
4) Modified temporal merge candidate derivation—enhancing the efficiency of the merge mode.
5) New default IBC merge candidates—enhancing the efficiency of the merge mode.

To provide a better understanding of the proposed techniques, palette mode and IBC are first reviewed in the following paragraphs.

Palette mode is designed according to the observation that the pixel values in a screen content frame usually concentrate on few color values. Fig. 1 shows an example of the palette mode. The encoder analyzes the pixels in a CU and determines



2	2	2	2
2	2	0	2
2	0	2	0
2	1	1	1

Palette	
Palette Index	Colour Value
0	50
1	80
2	210

Fig. 1. An example of palette mode

Manuscript received January 19, 2016; revised May 17, 2016 and July 15, 2016; accepted July 26, 2016. Date of publication August 16, 2016; date of current version December 9, 2016. Yu-Chen Sun is the corresponding author. This paper was recommended by Guest Editor R. A. Cohen.

The authors are with MediaTek Inc., Hsinchu 300, Taiwan (e-mail: ckey.sun@gmail.com; shih-ta.hsiang@mediatek.com; jungsun.kim@mediatek.com; ywchen0711@gmail.com; xiaozhong.xu@mediatek.com; sparkjj1985@gmail.com; shawmin.lei@mediatek.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2016.2598193

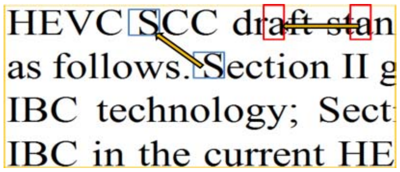


Fig. 2. Example of intra block copy (IBC).

several representative colors to construct a palette table; i.e., a color mapping table between the representative color values and color indexes. The palette table is signalled in the bitstream. The pixels with the pixel values close to the palette colors are quantized to the selected palette colors denoted by the corresponding palette indexes. The rest of the pixels are called escape pixels. The pixel values of the escape pixels are signalled directly. A special palette index value is reserved to represent the escape pixels. All palette indexes in the CU form a palette index map, which is transmitted to the decoder along with the escape pixel values. Note that in Fig. 1, to simplify the illustration, a pixel or a palette index is shown to correspond to only one value. However, in HEVC SCC Draft [3], a pixel or a palette index could represent three color component values (e.g., YCbCr or GBR).

Palette mode has been investigated for several years [5], [11]–[15]. During HEVC SCC standardization, several palette mode methods have been proposed [5], [13]–[15] and can be summarized into two parts: palette table coding tools and palette index map coding tools. In HEVC SCC, a palette is predictively coded and an index map is coded by a run-based approach. The palette table coding methods have been intensively studied in the past years [16]. Several techniques, e.g., prediction based on the last coded palette table [17], palette table predictor stuffing [18], [19], etc., have been proposed to improve coding efficiency by exploring the correlation between a palette table and its previous coded palette tables. A sophisticated palette table coding method has been adopted in HEVC SCC [16]. However, palette index map coding is relatively less optimized and still has room for improvement. For instance, in HEVC SCC, the palette indexes can only be predicted by other indexes within the same CU. However, the pixel values of a current CU are highly correlated to those of its neighboring boundary pixels. If this correlation can be explored, considerable coding gain improvement can be obtained. In this paper, a new technique, prediction across CU boundary, is proposed to address this issue. In addition, HEVC SCC only allows an index to be predicted by its above index. Yet, we have found that an index correlates with not only its above index, but also with all of its neighboring indexes, as well as the previous coded indexes with similar neighboring indexes. To address this issue, a transition copy technique is proposed.

For typical screen content, repetitive patterns can usually be found within the same picture as the example provided in Fig. 2. A new prediction mode, i.e., intra block copy (IBC) mode, is thus proposed to utilize this characteristic. In the IBC mode, a prediction unit (PU) is predicted from a previously reconstructed block within the same picture. Similar to motion compensated prediction, a displacement vector (called block vector or BV) is used to represent the relative displacement between the current PU and the reference block.

Since the IBC mode and the HEVC inter mode are very similar, in HEVC SCC, JCT-VC decided to unify the IBC operation and signalling with HEVC inter mode [7]–[10] and rename IBC as current picture referencing (CPR). The current (partially decoded) picture is treated as one of the reference picture for decoding the current slice. However, if strictly and totally following the unification principle is unnecessary, specific techniques can be developed for IBC to further improve coding efficiency. In this paper, three such methods are proposed. Screen content and natural videos can exhibit quite different statistical distributions of the motion vector differences (MVDs) after motion prediction. Thus, a new method is proposed for entropy coding the MVDs resulted from screen content. This universal entropy coding framework has already been adopted for coding palette index run values in HEVC SCC [37]. In this study, we further demonstrate the benefits of applying this method to block vector coding. The other two approaches for IBC are to improve temporal merge candidate derivation and to add default IBC merge candidates for enhancing the efficiency of the merge mode.

In this paper, the proposed improvements for palette mode and IBC are introduced in Sections II and III, respectively. In Section IV, the experimental results are shown to demonstrate the coding gains of individual techniques and a combination of all proposed techniques. Section V concludes this paper.

II. Improvements of Palette Mode

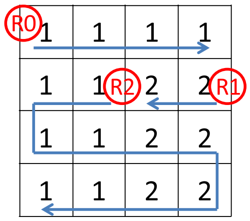
In this section, the proposed improvements of the palette mode are presented. The section begins with an overview of the palette mode in HEVC SCC. Then, two issues in HEVC SCC are addressed. Two new techniques, transition copy and prediction across CU boundary, are introduced to address the issues in Section II-B and Section II-C, respectively. The encoding algorithms are presented in Section II-D.

A. Overview of Palette Mode

In HEVC SCC palette mode, a flag is transmitted for each CU to signal whether the palette mode is used in the current CU. If the palette mode is utilized, the pixels with pixel values close to the palette colors are represented by the selected palette colors denoted by the palette indexes, where the corresponding palette colors are the representative colors in the current CU. The remaining pixels are presented by a special palette index value, denoted as an escape index, and their pixel values are signalled directly. Note that a color is a 3-value (e.g., YCbCr or GBR) vector.

Palette colors are described by a palette table, and encoded by palette table coding tools. Based on the palette table, a pixel of three color components is represented by a palette index. Palette indexes of all pixels in a CU form a palette index map and are encoded by palette index map coding tools.

Predictive coding is applied for the palette table. The decoder constructs a palette table predictor to predict the

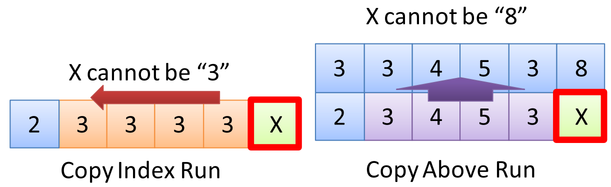


(a)

Run Index	Run Mode	Palette Index (derived from palette_index)	palette_index	Run of Pixels (derived from PaletteIndexRun)	PaletteIndexRun
R0	Copy Index	1	1	4	3
R1	Copy Index	2	1 (=2-1)	2	1
R2	Copy Above	N/A	N/A	10	9

(b)

Fig. 3. An example of palette index map coding: (a) an index map example, (b) the information to be encoded and the corresponding syntax in bit stream



(a) (b)

Fig. 4. Examples of redundant values of palette indexes

palette table of the current CU. In an early palette table coding method [13], [14], the predictor is the palette table of the neighboring CU. However, if the neighboring CU is not coded in palette mode, there is no valid predictor, so no prediction is applied in this case. As a result, the prediction of the palette table is inefficient. To solve this problem, it has been proposed to use the palette of the last palette coded CU as the predictor [17]. In other words, whether the neighboring CU is palette coded or not, a CU can always obtain a valid palette predictor as long as it is not the first palette-coded CU in a slice/tile. Furthermore, the prediction is found to be inefficient when a large CU uses a palette table of a small CU as palette table predictor because the small CU may contain much fewer colors than a large CU and fail to provide enough colors in the palette table predictor. It has been proposed to use not only the last coded palette table but also the colors in all previous coded palette tables to form a new predictor for the following CUs, so that a CU can acquire a palette table predictor with enough colors to perform predictions [18], [19]. More examples about palette table coding tools can be found in [16].

After the palette table is encoded, the pixels in a CU are represented by palette indexes corresponding to the colors in the palette. The indexes form an index map and are divided into several runs and then encoded in horizontal or vertical traverse scan order [21], [22]. In this paper, the horizontal scan order is assumed in the examples for illustration. Fig. 3 shows an example of the palette index map coding. In the figure, an index map of 16 indexes is encoded by three runs with horizontal traverse scan. There are two run modes: copy index mode and copy above mode. For each starting position of a run, a flag is transmitted to indicate which run mode is used.

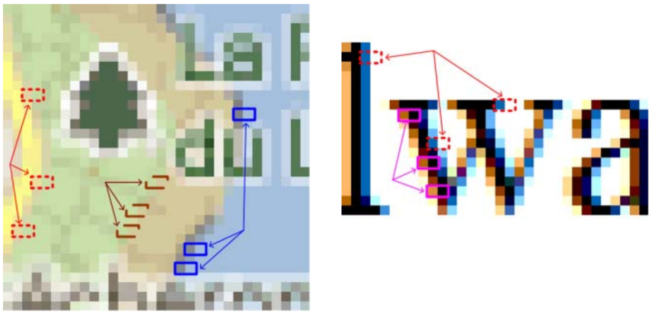
If the copy above mode is used, a run of pixels will copy the palette indexes from their above pixels, where the run value is derived from a piece of transmitted information, denoted as PaletteIndexRun. In the example in Fig. 3, a run, R2, is a copy above run with a run value is 10. Since the run value should be larger than zero, PaletteIndexRun is defined as PaletteIndexRun = run value–1. Thus, as shown in Fig. 3, PaletteIndexRun values for R0, R1, and R2 are 3, 1, and 9, respectively. Also, PaletteIndexRun is separated into a prefix syntax element and a suffix syntax element, which will be described in Section III-B.

If the copy index mode is used, a **palette_index_idc** syntax element is first signalled, followed by a piece of PaletteIndexRun information. A run of pixels share the same palette index, where the palette index and the run value are derived from the **palette_index_idc** and PaletteIndexRun, respectively. For instance, in Fig. 3, R0 and R1 are two copy index runs, whose (palette index, run value) are (1, 4) and (2, 2), respectively.

In the palette index map coding, redundant index removal is applied to improve coding efficiency. Fig. 4 illustrates how a redundant value of a palette index can be derived. In each of the two examples, X is the index to be encoded. Its previous run has two possibilities. When the previous run is a copy index run, as shown in the first example, the current index cannot be the same as the index of the previous pixel, i.e., 3 in Fig. 4(a). Because if the current index is the same as the index of previous copy index run, the current index shall be integrated into the previous copy index run. Therefore, the value 3 is redundant for the current palette index and can be discarded for entropy coding. On the other hand, when the previous run is a copy above run, the current index cannot be equal to the index of the above pixel, i.e., 8 in Fig. 4(b). Because if the current index is the same as the index of the above pixel, the current index shall be integrated into the previous copy above run. Therefore, 8 is a redundant value.

In both of the two aforementioned cases, a redundant value (denoted as adjustedRefPaletteIndex in the HEVC text specification) can be removed for the syntax element, **palette_index_idc**. In Fig. 3, R0 and R1 are copy index runs with palette index values of 1 and 2, respectively. For the **palette_index_idc** of the first run, there is no redundant value, so **palette_index_idc** is equal to the palette index, 1. For the **palette_index_idc** of the second run, the redundant value is the index of the previous pixel, 1. After removing the redundant value, **palette_index_idc** is 2-1 = 1.

Over the past few years, palette coding has been extensively studied. However, in comparison with palette table coding, palette index map coding is relatively less optimized and can be further improved. In the following subsections, two techniques are presented: transition copy and prediction across CU boundary. The former one can improve the performance of copy index mode, while the latter one can enhance the copy above mode.



(a) (b)

Fig. 5. An example of repeating color transition patterns in screen content

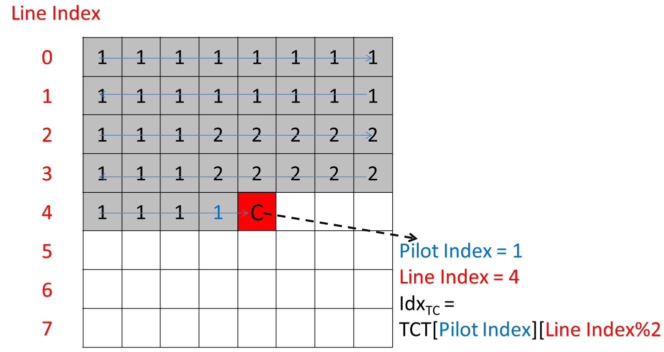


Fig. 6. An example of the transition copy (TC) prediction method

B. Transition Copy

In screen content, the occurrence of a color usually correlates with its neighboring colors. Fig. 5 illustrate this phenomenon using two examples. In Fig. 5(a), several repeating neighboring color pairs can be observed in the object boundary. For example, a color transition between gray and blue frequently occurs at the border between the land and sea. They are marked by the blue squares in Fig. 5(a). Other types of color transition patterns are also indicated by squares of different colors in Fig. 5(a). Besides, sub-pixel anti-alias rendering [23] is usually applied to the text region in screen content. This generates more repeating color transition patterns, such as the examples shown in Fig. 5(b): the red color often precedes the black color, and the blue color is usually near the white color.

Based on the above observation, a new color index prediction method, transition copy (TC), is proposed. TC is originally introduced in [24] and subsequently improved in [25]. It utilizes the transition correlation between a color and its neighboring colors to further increase coding efficiency. TC is an additional color index value prediction method for the copy index run mode. If an index is encoded by a copy index run, prior to encoding the index value, the encoder will derive a new index predictor, denoted as $\mathrm{Idx}_{\mathrm{TC}}$, using the TC method. If the prediction hits, the index value of the copy index run is inferred as $\mathrm{Idx}_{\mathrm{TC}}$, and the bits for the encoding index value (i.e., **palette_index_idc**) can be saved. If the prediction is unsuccessful, the index value will be encoded. Whether the prediction hits or misses, the decoder would get a decoded index for the current copy index run, and use the index to obtain the palette color value in the palette table to reconstruct the pixels in the current run.

An example of the TC palette index predictor is presented in Fig. 6. The current pixel, C, has a pilot palette index of the value 1. The shaded squares labeled with numbers are coded pixels. From the coded pixels, it can be found that when a palette index, 1, is coded in a left-to-right direction, the next palette index to the right is likely to be 2. This transition pattern (i.e., the index 1 => the index 2) is recorded in the TC table to enhance the efficiency of palette index prediction. This is the concept of TC.

When the current CU is the first CU of the slice, a TC table is reset at the first pixel. However, when the current CU is a non-first CU of the slice, the TC table inherits that of the previous CU. We call this inheriting property the "TC table propagation." The TC table is maintained and updated as the pixels in the CU are being coded. As shown in Table I, for each non-first-column pixel that is to be coded, the input of the TC table is called a pilot palette index, $\mathrm{Idx}_{\mathrm{pilot}}$, which is set to the left or right of the previous pixel depending on the scan direction. In addition, the output of the TC table is the TC palette index, $\mathrm{Idx}_{\mathrm{TC}}$. For a pilot palette index, $\mathrm{Idx}_{\mathrm{pilot}}$, two TC palette indexes, $\mathrm{TCT}[\mathrm{Idx}_{\mathrm{pilot}}][0]$ and $\mathrm{TCT}[\mathrm{Idx}_{\mathrm{pilot}}]\,[1]$, are recorded in the TC table. Whereas $\mathrm{TCT}[\mathrm{Idx}_{\mathrm{pilot}}][0]$ is likely to occur at a pixel position to the right of the pilot palette index relative to the causal area of the CU, $\mathrm{TCT}[\mathrm{Idx}_{\mathrm{pilot}}]\,[1]$ is likely to occur at a pixel position to the left of the pilot palette index. A pixel on the N_{th} row will choose $\mathrm{TCT}[\mathrm{Idx}_{\mathrm{pilot}}][\mathrm{N}_{\mathrm{th}}\%2]$ as the TC predictor.

TABLE I

AN EXAMPLE OF TC TABLE

Pilot Index	TC Predictor #0	TC Predictor #1
0	TCT[0][0]	TCT[0][1]
…	…	…
N_{TC}	$\mathrm{TCT}[\mathrm{N}_{\mathrm{TC}}][0]$	$\mathrm{TCT}[\mathrm{N}_{\mathrm{TC}}][1]$

The encoder and decoder will further refine the TC predictor. If the TC predictor is equal to the redundant index value mentioned in Section II-A (adjustedRefPaletteIndex) or is greater than or equal to the palette size, the TC predictor is useless since the index to be predicted should not equal these values. Hence, the encoder will modify the TC predictor into the smallest index that does not equal the redundant index value.

The refined predictor will be used to predict the current palette index. If the prediction misses, the pilot index and the current index, ($\mathrm{Idx}_{\mathrm{pilot}}$, the current palette index), become a new transition pattern that updates the TC table. In addition, the current index is sometimes found to be equal to its pilot index. Such a pattern is inefficient. Therefore, we only update the TC table if the current index is not the same as the pilot color index. To update a TC table, the encoder and decoder replace $\mathrm{TCT}[\mathrm{Idx}_{\mathrm{pilot}}][\mathrm{N}_{\mathrm{th}}\%2]$ with the current palette index and $\mathrm{TCT}[\text{the current palette index}][(\mathrm{N}_{\mathrm{th}}+1)\%2]$ with $\mathrm{Idx}_{\mathrm{pilot}}$.

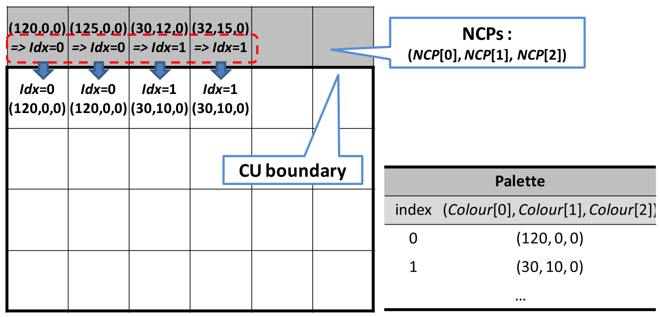


Fig. 7. An example of prediction across CU boundary

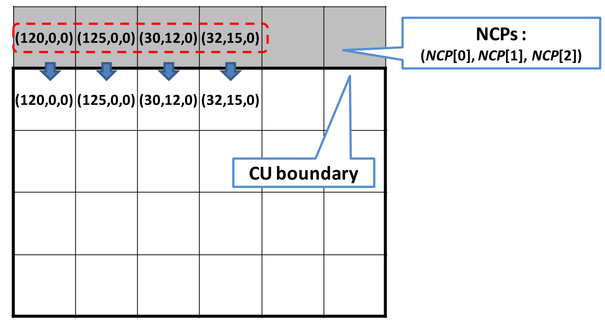


Fig. 8. An example of simplified prediction across CU boundary, where the pixel values outside the current CU are directly used to represent the pixel values inside the current CU without transforming pixel values outside the current CU to palette indexes

The two updated values correspond to the same transition pattern yet differ in the scan directions.

Note that, because the refined TC predictor is always different from the redundant index (adjustedRefPaletteIndex), a second redundant index ($\text{adjustedRefPaletteIndex}_2$) is recorded and set to equal the refined TC predictor. In this way, if an index value is being coded in an index run mode, more redundancy can be removed to represent the index value. In other words, the to-be-encoded index value of the index run mode can neither be the same as the first reference index (adjustedRefPaletteIndex), nor the second reference index ($\text{adjustedRefPaletteIndex}_2$).

C. Prediction Across CU Boundary

In HEVC SCC, the run of the first column cannot be a copy above run because the above pixel position is outside the current CU and no corresponding indexes can be used for prediction in a copy above run. Moreover, the run type is not signalled and forced to be copy index run. In this scheme, the correlation between the current CU and a neighboring CU remains unused. If the correlation can be further utilized, coding efficiency can be improved.

In this subsection, we propose to use the partially decoded pixels (i.e., before deblocking) of the nearest above row (or the nearest left column depending on the scan order) in a neighboring CU to generate the index prediction sources [26]–[28]. These pixels are referred to as the neighboring CU pixels (NCPs). For example, Fig. 7 shows the NCPs marked in gray. An NCP, NCP_i, has three decoded pixel values, (NCP_i [0], NCP_i [1], *NCPi* [2]), corresponding to YCbCr or GBR.

To reconstruct the palette indexes of the current CU, the decoder transforms each NCP's values into a palette index. If the current CU is located at the slide/picture boundary, i.e., NCPs are outside of the slide/picture boundary, the palette indexes of NCPs are inferred as 0. Otherwise, the decoder assigns indexes for NPCs that minimizes the difference between the NCP's values and the pixel values corresponding to the palette index in the current CU's palette table. Assume that there are N colors in the palette table, e.g., $Color_0$, $Color_1$,... $Color_{N-1}$, where a color, $Color_j$, has three pixel values, ($Color_j$ [0], $Color_j$ [1], $Color_j$ [2]), and a palette index j. For an NCP, NCP_i, its transformed index, Idx_i, is defined as

$$Idx_i = \begin{cases} 0, & \text{If the current CU is located} \\ & \text{at slide/picture boundary.} \\ \underset{0\le n<N}{\arg\min}\left(\sum_{k=0}^{2} |NCP_i[k] - Colour_n[k]|\right), & \\ & \text{Otherwise.} \end{cases} \tag{1}$$

Fig. 7 shows an example of the palette of the current CU. The decoder uses the palette to transform the NCP's values into palette indexes. The resulting indexes, *Idx*, of the NCPs from (1) are listed underneath the pixel values in Fig. 7. The copy-above run mode at the CU boundary is also illustrated according to the proposed scheme. To reconstruct a palette index for the first row of the current CU, if the index is encoded by a copy-above run mode, the decoder reconstructs the index from the corresponding NCP index.

In the aforementioned method, the computation burden might be increased if the decoder is required to transform the NCP's partially decoded sample values into a palette index by (1). As a solution, instead of deriving an index from the NCP, we directly use the NCP's partially decoded pixel values to reconstruct pixels in the current CU. Therefore, the conversion process, (1), can be removed entirely. Fig. 8 provides an example. To reconstruct the sample values of the first row of the current CU, if these pixels' indexes are encoded by a copy-above run, the decoder reconstructs the pixels from the NCP pixel values instead of the NCP-derived indexes. The results in [28] show that the coding loss of the simplification is minor and indicate that the simplified method achieves good performance-complexity trade-off.

In the simplified method (or called copy pixel method), the decoder does not derive indexes for NCPs. Therefore, the pixels reconstructed by NCPs are not assigned palette indexes. Consequently, the redundant index removal can be applied for coded indexes partially. If the redundant index for a copy-index run mode is derived from an index-less pixel, the redundant index removal function will be disabled [28]. In addition, as a result of combining the copy pixel value method with TC prediction, if the pixel of the pilot index is derived from an NCP (i.e., no index value exists in the pilot pixel position), TC prediction will be disabled. Therefore, when the simplified method and TC prediction are combined, a pixel could be encoded by a copy index run or a copy above run. If the pixel is encoded by a copy index run, the pixel will always be reconstructed by the decoded index whether the TC prediction hits or misses. If the pixel is encoded by a copy above run, there are two scenarios: the decoder would reconstruct the current pixel by directly copying the decoded pixel value of the above pixel if the above pixel is a NCP or reconstructed by a NCP; otherwise, the decoder will copy the index of the above pixel to reconstruct the current pixel.

It is worth mentioning that, with the proposed copy pixel method, pixel reconstructions are collaboratively conducted by the NCPs and the palette table. In comparison, in HEVC SCC, all pixels are reconstructed by the palette table. Therefore, we can further improve the encoding optimization method as described in the following subsection.

D. Encoding Algorithm

A palette encoder achieves the best coding performance by finding the optimal pair of palette table and corresponding palette index map. The encoding algorithm proposed in this paper involves three steps: 1) deriving an initial palette table and a corresponding palette index map; 2) determining the run mode of each palette index in the palette index map; 3) refining the palette table based on the run mode assignment.

The first two steps are similar to the algorithm used in the HEVC SCC reference software [29]. The encoder first calculates the color histogram of the pixels in a CU and then chooses N most popular colors to form a palette table. In addition, if a chosen color is similar to a color in the palette table predictor, the chosen color will be replaced by the similar color in the palette table in order to save bits used for describing the color. After obtaining the initial palette table, the encoder transforms all pixels into palette indexes, which are subsequently encoded in several runs. Each index may be encoded by the copy index run or the copy above run. To determine the run modes, the encoder traces all palette indexes according to the scan order. Starting from the first index, the encoder calculates the maximal run length of the two run options and their rate-distortion (R-D) costs. To evaluate the R-D costs of the two run modes, Transition Copy and Prediction across CU Boundary are integrated in the copy index mode and the copy above mode, respectively. After selecting the run mode with the lower R-D cost, the encoder moves the starting position to the next run and repeats the procedure of choosing the better run mode. The run mode assignment is done after all of the palette indexes in the CU have been visited.

In the HEVC SCC reference software, the palette table generation and the run mode assignment are completed by the two steps mentioned above. Here, we propose to add an additional step based on the observation that, with prediction across CU boundary, a part of the pixels is reconstructed by NCPs. Because these pixels are not affected by the colors in the palette table, we can exclude them and re-calculate the optimal palette colors for the remaining pixels. The optimal palette colors, *Color'* $_0$, *Color'* $_1$,... *Color'* $_{N-1}$, are derived to minimize the quantization error, E:

$$E = \sum_{n=0}^{N-1} \sum_{i=0}^{M} \left(a_{in} \times \sum_{k=0}^{2} \left| P_i[k] - Colour'_n[k] \right|^2 \right) \quad (2)$$

where P_i is the color value of pixel i, and M is the amount of pixels in the CU. The Boolean variables, a_{in}, represent how pixel i is constructed: a_{in} equals 1 if pixel I is reconstructed by *Color'* $_n$; otherwise, a_{in} equals 0. Note that if a pixel i is reconstructed by the NCPs instead of the palette table, a_{in} equals 0 for all n.

According to the initial palette table and the run mode assignment from the first two steps, we can calculate a_{in}. Then, the refined palette color values, *Color'* $_n$ $[k]$, can be derived to minimize the error in (2):

$$Color'_n[k] = \frac{\sum_{i=0}^{M} (a_{in} \times P_i[k])}{\sum_{i=0}^{M} a_{in}}. \quad (3)$$

To evaluate the performance of the proposed palette refinement step, we calculate the BD-rate difference between the related results generated by the software with and without the refinement step. The experiments are conducted under the common test conditions defined by JCT-VC [43], which will be introduced in section IV. The results are summarized in Table II and show that the palette refinement step can provide 0.2%–0.3% average bit saving among all test sequences.

TABLE II
BD-RATE PERFORMANCE OF THE PROPOSED PALETTE REFINEMENT TECHNIQUE. THE ANCHOR IS THE SOFTWARE WITHOUT THE REFINEMENT TECHNIQUE. THE TEST IS THE SOFTWARE WITH THE REFINEMENT TECHNIQUE. NEGATIVE VALUES MEAN BIT RATE DECREASES (CODING GAINS)

	Palette Refinement		
	AI	**RA**	**LB**
RGB, text & graphics with motion, 1080p & 720p	-0.3%	-0.3%	-0.5%
RGB, mixed content, 1440p & 1080p	-0.4%	-0.2%	-0.1%
RGB, Animation, 720p	0.0%	0.0%	0.0%
RGB, camera captured, 1080p	0.0%	0.0%	-0.1%
YUV, text & graphics with motion, 1080p & 720p	-0.5%	-0.3%	-0.5%
YUV, mixed content, 1440p & 1080p	-0.4%	-0.2%	-0.1%
YUV, Animation, 720p	0.0%	0.0%	0.0%
YUV, camera captured, 1080p	0.0%	0.0%	0.0%
Average	-0.3%	-0.2%	-0.3%

III. IMPROVEMENTS OF INTRA BLOCK COPY

A. Overview of Intra Block Copy

In HEVC SCC, IBC has been implemented in a similar way as the HEVC motion compensation process. The current decoded picture is treated as one of the reference pictures for decoding the current slice. This current reference picture is placed in the reference picture list. When IBC is employed for a PU, its reference picture is the current picture. When a frame is being encoded, if the current picture is the only reference

TABLE III
TEST SEQUENCES IN THE COMMON TEST CONDITIONS OF HEVC SCC

Resolution	Sequence	Bit Depth	Category	Fps
1920x1080	FlyingGraphics	8-bit	TGM	60
1920x1080	Desktop	8-bit	TGM	60
1920x1080	Console	8-bit	TGM	60
1280x720	Web_browsing	8-bit	TGM	30
1280x720	Map	8-bit	TGM	60
1280x720	Programming	8-bit	TGM	60
1280x720	SlideShow	8-bit	TGM	20
2560x1440	Basketball_Screen	8-bit	MC	60
2560x1440	MissionControlClip2	8-bit	MC	60
1920x1080	MissionControlClip3	8-bit	MC	60
1280x720	Robot	8-bit	ANI	30
1920x1080	EBURainFruits	10-bit	CC	50
1920x1080	Kimono1	10-bit	CC	24

picture in the reference picture list, the encoding process is equivalent to intra-encoding with IBC. This reference picture is inserted into the last position in the reference picture list and is marked as "used for long-term reference" during the decoding of the current picture.

Although unifying IBC and HEVC inter coding can simplify the design, the properties of IBC and HEVC inter may not be exactly the same. In this section, three approaches are introduced: improved MV coding, modification of temporal merge candidate derivation, and default IBC merge candidates.

B. Improved MV Coding

This subsection proposes to utilize a new entropy method for coding the motion vector difference (MVD). This method represents an unsigned source sample value in a binary form by signalling its most significant bit (MSB) index followed by its refinement bits. The MSB index for the sample value equal to 0 is assigned to -1.

The prefix part of the resulting codeword represents the MSB index plus 1, denoted by **msb_id_plus1**. The prefix part msb_id_plus1 of an unsigned source sample value x is given by

$$\text{msb\_id\_plus1} = \begin{cases} \text{Floor}\,(\text{Log2}\,(x)) + 1, & x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where Log2(·) represents the base-2 logarithm and Floor(·) returns the largest integer less than or equal to the input value.

The suffix part **refinement_bits** represents the value derived from the refinement bits and is only present if msb_id_plus1 is greater than 1. The suffix part refinement_bits of an unsigned source sample value x is given by

$$\text{refinement\_bits} = x\&((1 \ll (\text{msb\_id\_plus11})) - 1), \tag{5}$$

where "&" represents the bit-wise AND operator. The decoded syntax value x can be derived by

$$x = \begin{cases} \text{msb\_id\_plus1}, \text{msb\_id\_plus1} < 2 \\ (1 \ll (\text{msb\_id\_plus1} - 1)) + \text{refinement\_bits, otherwise.} \end{cases} \tag{6}$$

For representing a signed source sample, the proposed method can be applied to coding the absolute value of the source sample together with the additionally coded sign bit. This new entropy coding method has been proposed for coding several new syntax elements employed by the HEVC SCC coding tools and demonstrated coding efficiency gains [30]–[38]. It has been adopted by the HEVC SCCC extension for run-length coding the palette index map [37].

For entropy coding the MVD resulted from motion vector prediction, the proposed method can be applied to each vector component of the MVD. For representing the prefix part msb_id_plus1, the propose method first codes a syntax flag **msb_p1_gr0_flag** to indicate if the prefix value is greater than 0. If the flag is true, then the remaining prefix value is tested against a threshold **msb_thre**. A second syntax flag **msb_p1_grT_flag** is next coded to indicate if the prefix value msb_id_plus1 is greater than the threshold msb_thre. If the flag is true, then the syntax element **msb_minusT** equal to (msb_id_plus1–1–msb_thre) is coded in the unary code, otherwise, the syntax element **thre_minus_msb_p1** equal to (msb_thre–msb_id_plus1) is coded in the truncated unary code with the maximum value equal to (msb_thre–1). The decoded prefix msb_id_plus1 can be derived by

$$\text{msb\_id\_plus1} = \begin{cases} 0, & \text{msb\_p1\_gr0\_flag} = 0 \\ \text{msb\_minusT} + \text{msb\_thre} + 1, & \text{msb\_p1\_grT\_flag} = 1 \\ \text{msb\_thre} - \text{thre\_minus\_msb}_{\text{p}}1, & \text{otherwise.} \end{cases} \tag{7}$$

The threshold msb_thre is coded in the slice header for each vector component. In our implementation, the value of msb_thre is determined by the histogram of coded sample values from the previous frames corresponding to the same temporal level. The suffix part refinement_bits is binarized in a fixed length code with the code length equal to (msb_id_plus1–1).

Three contexts are allocated for entropy coding the syntax flag msb_idx_p1_gr0_flag[c], where c = 0 or 1 indicates the index of a vector component. One single context is dedicated to coding msb_p1_gr0_flag[0]. Two contexts are employed for coding msb_p1_gr0_flag [1] conditioned on the coded value of msb_idx_p1_gr0_flag[0]. In this way, statistical dependency between two vector components can be further exploited. The same method was also independently developed in [39]. The syntax flag msb_p1_grT_flag for a vector component is coded using one single context. The bin strings resulted from the syntax elements msb_minusT and thre_minus_msb_p1 are coded in the CABAC mode if the current bin index is less than a pre-determined bypass threshold value, in the bypass mode, otherwise. The context selection is determined by the current bin index and vector component index. Separate context sets

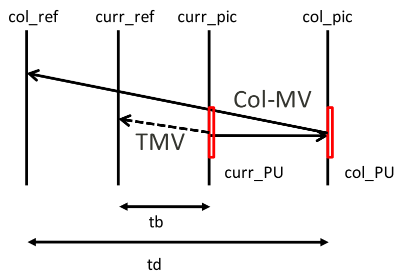


Fig. 9. Illustration of motion vector scaling for temporal merge candidate

are allocated for entropy coding the MVDs corresponding to inter-frame and intra-frame prediction.

C. Modification of Temporal Merge Candidate Derivation

In HEVC merge mode, a temporal merge candidate is derived from the MV of the collocated PU in the collocated picture as illustrated by the dotted line in Fig. 9. The temporal merge candidate (TMV) is derived by using the following equation:

$$\mathrm{TMV} = \mathrm{ColMV} \times \frac{\mathrm{tb}}{\mathrm{td}} \tag{8}$$

where the ColMV is the MV of the collocated PU in the collocated picture, the tb is the picture order count (POC) difference between the reference picture of the current picture and the current picture, and the td is the POC difference between the reference picture of the collocated picture and the collocated picture. In temporal merge candidate derivation, if the reference picture of the collocated picture is marked as the long-term reference picture and the reference picture of the current picture is mark as the short-term reference picture, the temporal merge candidate is not available.

In HEVC SCC, the reference picture for IBC mode is the current picture, which is a long-term reference picture. If the collocated PU is coded in IBC mode and the reference picture of the current picture is a short-term picture, the MV of the collocated IBC coded PU cannot be used to derive the temporal merge candidate because the reference picture of the collocated picture is the long-term reference picture and the reference picture of the current picture the short-term reference picture. The coding efficiency of the merge mode is compromised.

In this subsection, the MVs of the IBC coded PU are proposed to be utilized in temporal merge candidate derivation. In temporal merge candidate derivation, if the collocated PU is coded in IBC mode, the temporal merge candidate is set as an IBC candidate. The MV is set equal to the MV of the collocated PU and the reference picture index is set equal to the index of the reconstructed current reference picture.

D. Default IBC Merge Candidates

In HEVC, when the merge candidate list is not full, the zeroMV candidates are used to fill the empty entries in the merge list. For the zeroMV candidate, the MV is set to (0, 0) and the reference picture index is set to 0. Such a merge candidate list construction in HEVC works fine when dealing with camera-captured content. In screen content, the efficiency of the use of IBC mode indicates that spatial correlation among neighboring pixels is much higher than in camera-captured content. In other words, there is a good chance that the current block is very similar to its close neighbors. When setting the default merge candidates in the context of SCC, the popularity of IBC usage should be considered.

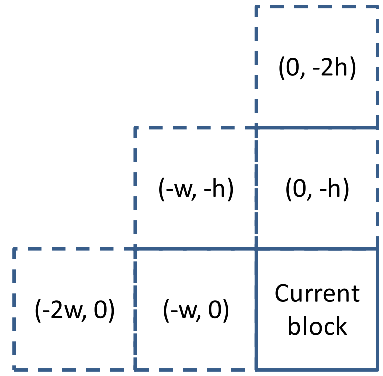


Fig. 10. Illustration of default IBC merge candidates

In this subsection, it is proposed to add a few predefined IBC block vectors to the list of merge candidates to fill the empty merge entries [40], [41]. Considering that the value of a block vector has some constraints, the candidates of the MV equal to (–w, 0), (–2w, 0), (0, –h), (0, –2h), and (–w, –h) with the reference picture index equal to the index of the reconstructed current reference picture are used to replace the zeroMV candidates, where w and h are the width and height of the current CU, respectively. The design spirit of using these values is to include the most commonly used neighboring positions to predict the current block. In Fig. 10, an illustration of these candidates is shown.

IV. Experimental Results

A. Experiment Settings

The proposed techniques have been integrated into the HEVC SCC reference software, SCM-4.0 [29], which includes all HEVC SCC tools, such as adaptive color transform, palette mode, IBC, and so on. To accommodate the proposed palette techniques, the syntax grouping in [42] is removed, which does not have any coding efficiency impact. The experiments are conducted under the common test conditions defined by JCT-VC [43]. The full frame search configuration for IBC is used. A set of non-camera captured as well as camera captured video sequences are tested in both RGB 4:4:4 and YUV 4:4:4 color formats. Table II shows all test sequences: seven sequences are selected to represent the most common screen content videos, referred as “text and graphics with motion (TGM) class”; three sequences are selected containing a mixture of both camera-captured content and text/graphics, referred as “mixed content (MC) class”; one animation (ANI) sequence and two camera-captured (CC) video sequences are also tested. Fig. 11 shows some example frames of each category. The coding performance is measured by BD-rate [44].

(a) Desktop (Text & Graphics with Motion)

(b) MissionControlClip3 (Mixed Content)

(c) Robot (Animation)

(d) Kimono1 (Camera Captured)

Fig. 11. Example frames of each sequence category.

TABLE IV

BD-RATE PERFORMANCE OF EACH INDIVIDUAL PALETTE TECHNIQUE. THE ANCHOR IS SCM-4.0. THE COLUMNS, "(1) TRANSITION COPY", "(2) PREDICTION ACROSS CU", AND "(1) + (2)", PRESENT SCM-4.0 COMBINING TRANSITION COPY, PREDICTION ACROSS CU, AND BOTH OF THE PALETTE TECHNIQUES, RESPECTIVELY. NEGATIVE VALUES MEAN BIT RATE DECREASES (CODING GAINS). THE FULL FRAME SEARCH CONFIGURATION FOR IBC IS USED

	(1) Transition Copy			**(2) Prediction across CU**			**(1) + (2)**		
	AI	**RA**	**LB**	**AI**	**RA**	**LB**	**AI**	**RA**	**LB**
RGB, text & graphics with motion, 1080p & 720p	-1.0%	-0.7%	-0.5%	-1.8%	-1.3%	-1.1%	-2.8%	-1.9%	-1.5%
RGB, mixed content, 1440p & 1080p	-0.2%	-0.1%	-0.3%	-0.8%	-0.5%	-0.2%	-0.9%	-0.7%	-0.5%
RGB, Animation, 720p	0.0%	0.1%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	-0.1%
RGB, camera captured, 1080p	0.0%	0.0%	-0.1%	0.0%	0.0%	-0.1%	0.0%	0.0%	0.0%
YUV, text & graphics with motion, 1080p & 720p	-0.9%	-0.6%	-0.4%	-2.4%	-1.6%	-1.3%	-3.2%	-2.2%	-1.7%
YUV, mixed content, 1440p & 1080p	-0.3%	-0.2%	-0.2%	-1.0%	-0.8%	-0.5%	-1.3%	-0.9%	-0.8%
YUV, Animation, 720p	0.0%	0.0%	0.1%	-0.1%	-0.1%	-0.2%	-0.1%	0.0%	-0.1%
YUV, camera captured, 1080p	0.0%	0.0%	0.0%	0.0%	-0.1%	0.0%	0.0%	0.0%	0.0%
Average	-0.6%	-0.4%	-0.3%	-1.3%	-0.9%	-0.8%	-1.9%	-1.3%	-1.0%
Enc Time[%]	105%	104%	103%	106%	101%	101%	112%	106%	105%
Dec Time[%]	104%	105%	104%	103%	103%	104%	105%	105%	105%

To simplify the result tables, in the section, only BD-rate results of Y/G component are shown. The results of other two components are similar, so they are omitted for brevity. The simulations are performed using the 64bit Linux platform with Xeon 5160 3.0GHz CPUs.

B. Results of Individual Techniques

Two proposed palette techniques are evaluated on SCM-4.0. The coding gains (i.e., BD-rate decreases, negative values) of each technique can reflect the degree of improvement. The results are shown in Table IV. For "YUV, text & graphics with motion, 1080p & 720p sequences" under all intra (AI) configuration, 0.9% and 2.4% BD-rate saving could be observed for (1) transition copy and (2) prediction across CU boundary, respectively, while the combination of the two palette improvements can improve coding efficiency by 3.2%.

For IBC improvements, the results are shown in the Table V. Note that "modification of temporal merge candidate derivation" is an inter-coding tool, so the results are the same as the anchor under AI configuration. Under random access (RA) configuration, 2.6%, 0.5%, and 0.4% BD-rate saving could be observed for (3) improved MV coding, (4) modification of temporal merge candidate derivation, and (5) default IBC merge candidates, respectively, for "YUV, text & graphics with motion, 1080p & 720p sequences". And, the

TABLE V

BD-RATE PERFORMANCE OF EACH INDIVIDUAL IBC TECHNIQUE. THE ANCHOR IS SCM-4.0. THE COLUMNS, " (3) MV CODING", " (4) TMC DERIVATION", " (5) DEFAULT IBC", AND " (3) + (4) + (5) ", PRESENT SCM-4.0 COMBINING IMPROVED MV CODING, MODIFICATION OF TEMPORAL MERGE CANDIDATE DERIVATION, DEFAULT IBC MERGE CANDIDATES, AND ALL OF THE IBC TECHNIQUES, RESPECTIVELY. NEGATIVE VALUES MEAN BIT RATE DECREASES (CODING GAINS). THE FULL FRAME SEARCH CONFIGURATION FOR IBC IS USED

	(3) MV Coding			(4) TMC Derivation			(5) Default IBC			(3) + (4) + (5)		
	AI	RA	LB	AI	RA	LB	AI	RA	LB	AI	RA	LB
RGB, text & graphics with motion, 1080p & 720p	-1.8%	-2.3%	-2.3%		-0.6%	-0.6%	-0.5%	-0.3%	0.0%	-2.2%	-3.0%	-3.1%
RGB, mixed content, 1440p & 1080p	-0.9%	-0.8%	-0.6%		-0.2%	-0.3%	-0.1%	-0.2%	-0.1%	-0.9%	-1.1%	-1.1%
RGB, Animation, 720p	-0.1%	-0.2%	-0.2%		0.0%	0.0%	0.0%	0.0%	0.1%	-0.1%	-0.2%	-0.2%
RGB, camera captured, 1080p	0.0%	-0.3%	-0.3%		0.0%	0.0%	0.0%	0.0%	0.0%	-0.1%	-0.3%	-0.2%
YUV, text & graphics with motion, 1080p & 720p	-2.1%	-2.6%	-2.6%		-0.5%	-0.6%	-0.8%	-0.4%	-0.2%	-2.8%	-3.4%	-3.5%
YUV, mixed content, 1440p & 1080p	-1.2%	-0.9%	-0.7%		-0.2%	-0.4%	-0.6%	-0.5%	-0.1%	-1.6%	-1.5%	-1.3%
YUV, Animation, 720p	-0.2%	-0.2%	0.0%		-0.1%	-0.1%	-0.1%	-0.1%	0.1%	-0.3%	-0.3%	-0.2%
YUV, camera captured, 1080p	-0.1%	-0.3%	-0.2%		0.0%	0.0%	-0.1%	0.0%	0.0%	-0.1%	-0.3%	-0.1%
Average	-1.3%	-1.6%	-1.5%		-0.3%	-0.4%	-0.5%	-0.3%	-0.1%	-1.7%	-2.1%	-2.1%
Enc Time[%]	102%	96%	96%		98%	99%	90%	98%	99%	91%	94%	94%
Dec Time[%]	101%	103%	101%		103%	101%	97%	101%	100%	99%	101%	103%

TABLE VI

BD-RATE PERFORMANCE OF THE COMBINATION OF ALL PROPOSED TECHNIQUES. NEGATIVE VALUES MEAN BIT RATE DECREASES (CODING GAINS). THE FULL FRAME SEARCH CONFIGURATION FOR IBC IS USED

	(1) + (2) + (3) + (4) + (5)		
	AI	RA	LB
RGB, text & graphics with motion, 1080p & 720p	-4.6%	-4.6%	-4.3%
RGB, mixed content, 1440p & 1080p	-1.5%	-1.6%	-1.4%
RGB, Animation, 720p	-0.1%	-0.2%	-0.3%
RGB, camera captured, 1080p	-0.1%	-0.3%	-0.3%
YUV, text & graphics with motion, 1080p & 720p	-5.4%	-5.2%	-4.9%
YUV, mixed content, 1440p & 1080p	-2.4%	-2.2%	-1.8%
YUV, Animation, 720p	-0.3%	-0.2%	-0.2%
YUV, camera captured, 1080p	-0.1%	-0.4%	-0.1%
Average	-3.2%	-3.2%	-2.9%
Enc Time[%]	102%	96%	96%
Dec Time[%]	101%	103%	101%

TABLE VII

MODE DISTRIBUTION UNDER AI CONFIGURATION. THE SEQUENCE, FLYINGGRAPHICS, IS ENCODED BY SCM-4.0 WITH AND WITHOUT THE PROPOSED TECHNIQUES. THE NUMBERS IN THE TABLE ARE THE PERCENTAGE OF PIXELS ENCODED BY EACH MODE

		IBC	Palette	HEVC Intra
SCM-4.0	QP=22	74.1%	17.9%	8.0%
	QP=37	73.6%	11.7%	14.7%
SCM-4.0 + (1) + (2) + (3) + (4) + (5)	QP=22	77.0%	19.9%	3.1%
	QP=37	78.3%	15.2%	6.5%

combination of all IBC improvements can improve coding efficiency by 3.4%.

C. Results of Combining All Proposed Techniques

The combination of all proposed techniques is also evaluated. As shown in Table VI, although almost no BD-rate difference can be observed for "camera captured sequences," the BD-rate decreases are 5.4%, 5.2%, and 4.9% for "YUV, text & graphics with motion, 1080p & 720p sequences" under AI, RA, and LB, respectively. The proposed techniques perform well in pure screen content sequences. And, the techniques also can achieve around 2% BD-rate decreases for "mixed content sequences."

To further examine how the proposed techniques affect the selection of the CU modes, the distribution of the coded CU modes is summarized in Table VII and Table VIII. We choose a complex TGM sequence, FlyingGraphics, to conduct experiments. The sequence is encoded by SCM-4.0

TABLE VIII

MODE DISTRIBUTION UNDER LB CONFIGURATION. THE SEQUENCE, FLYINGGRAPHICS, IS ENCODED BY SCM-4.0 WITH AND WITHOUT THE PROPOSED TECHNIQUES. THE NUMBERS IN THE TABLE ARE THE PERCENTAGE OF PIXELS ENCODED BY EACH MODE. NOTE THAT ONLY PIXELS IN INTER SLICES ARE COUNTED

		IBC	**Palette**	**HEVC Intra**	**HEVC Inter**
SCM-4.0	QP=22	37.1%	5.9%	3.6%	53.4%
	QP=37	9.2%	1.8%	1.3%	87.7%
SCM-4.0 + (1) + (2) + (3) + (4) + (5)	QP=22	48.0%	7.7%	0.7%	43.6%
	QP=37	15.3%	3.4%	0.4%	80.9%

with and without the proposed techniques. Table VII shows the results under AI configuration. Table VIII presents the results under low-delay B (LB) configuration, whereby only the inter slices are counted, that is, the intra slices is excluded. In Table VII, it can be observed that IBC represents the majority of the modes. The proposed techniques can increase the proportion of IBC by 3 $\sim$ 4%. Palette mode is the second most frequently used, and its proportion can also be increased by 3% with our method. Similarly, when applied on inter slices, proportions of both the palette mode and IBC can be increased (Table VIII).

D. Complexity Analysis

Complexity of the proposed IBC improvements can be summarized as follows: The proposed method for entropy coding MVDs can increase the max number of context-coded bins and the number of modeling contexts. For generating the experimental results in Table IV, our method can use up to four context-coded bins for coding each MVD vector component and employs 26 modeling contexts for coding all MVD vectors. However, the max number of context-coded bins and the number of modeling contexts utilized by our method can be reduced without significant performance sacrifice [38]. Modified temporal merge candidate derivation and new default IBC merge candidates still follow the mechanism of the merge mode in HEVC SCC. However, the former utilizes the MVs of the IBC coded PU in temporal merge candidate derivation and, the latter adds new merge candidates. There is almost no increase in computational complexity and buffer requirement for the proposed IBC improvements. The encoding/decoding run times listed in Table V confirm this: the increase in decoding run time is within 3%, while the encoding run time is decreased. The reason for the decrease in encoding time may be that the proposed tools improve the prediction efficiency of IBC. The prediction efficiency improvement benefits from the fast rate-distortion (RD) optimization algorithm in SCM-4.0 in that the fast RD optimization can be terminated early and the checking of the RD cost for intra mode can be bypassed.

For the proposed palette improvements, prediction across CU boundary just allows the pixel reconstruction process of the palette mode to access pixels outside of the CU. And, the buffer of those pixels can be shared with the intra prediction module. Transition copy requires the decoder to maintain a TC table. The operation is simple, including looking up the table to find the TC predictor and updating it. And, the operation can be performed in the reconstruction stage, not the parsing stage, so that it does not increase parsing complexity. For memory consumption, TC requires a buffer to store a TC table. The size of the TC table is ($2^{*}\mathrm{N}_{\mathrm{tx}}^{*}\mathrm{N}_{\mathrm{Maxidxbitdepth}}$) bits, where N_{tx} and $\mathrm{N}_{\mathrm{Maxidxbitdepth}}$ are the maximum number of pilot indexes and maximum bit depth of indexes, respectively. Under the common test conditions, N_{tx} and $\mathrm{N}_{\mathrm{Maxidxbitdepth}}$ are set as 31 and 5, so the size of the TC table is 310 bits. From the run time information in Table IV, it can be found that the decoding run time increases are within 5% while the encoding time increases are 12%, 6%, and 5% under AI, RA, and LB configurations, respectively.

The encoding/decoding times of the combination of all proposed improvements are summarized in Table VI and the increases in encoding/decoding times are all less than 3%. It implies a very good gain-complexity trade-off.

V. CONCLUSION

In this paper, five techniques are proposed to improve the coding efficiency of the palette mode and intra block copy in HEVC SCC, including (1) transition copy, (2) prediction across CU boundary, (3) advanced MV coding, (4) a modified temporal merge candidate derivation, and (5) new default IBC merge candidates. Experimental results show that compared with HEVC SCC, the proposed techniques can achieve up to 5.4%, 5.2%, and 4.9% BD-rate savings under all intra (AI), random access (RA), and low-delay B (LB) common test conditions, respectively. Since screen content coding has been widely used for many applications, improvements are desired and will be investigated in the future. The proposed approaches represent potential directions for improving the current HEVC SCC and future codec developments.

REFERENCES

[1] Y. Lu, S. Li, and H. Shen, "Virtualized screen: A third element for cloud–mobile convergence," *IEEE Multimedia Mag.*, vol. 18, no. 2, pp. 4–11, Feb. 2011.

[2] R. Joshi *et al.*, *Screen Content Coding Test Model 1 (SCM 1)*, document JCTVC-Q1014, Joint Collaborative Team on Video Coding, 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[3] R. Joshi and J. Xu, *HEVC Screen Content Coding Draft Text 1*, document JCTVC-R1005, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[4] L. Zhang *et al.*, *SCCE5 Test 3.2.1: In-Loop Color-Space Transform*, document JCTVC-R0147, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[5] P. Onno, X. Xiu, Y.-W. Huang, and R. Joshi, *Suggested Combined Software and Text for Run-Based Palette Mode*, document JCTVC-R0348, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[6] M. Budagavi and D.-K. Kwon, *AHG8: Video Coding Using Intra Motion Compensation*, document JCTVC-M0350, Joint Collaborative Team on Video Coding, 13th Meeting, Incheon, South Korea, Apr. 2013, pp. 18–26.

[7] X. Xu, S. Liu, and S. Lei, *On Unification of Intra Block Copy and Inter-Picture Motion Compensation*, document JCTVC-Q0132, Joint Collaborative Team on Video Coding, 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[8] B. Li and J. Xu, *Non-SCCE1: Unification of Intra BC and Inter Modes*, document JCTVC-R0100, 18th Meeting, Joint Collaborative Team on Video Coding, Sapporo, Japan, Jun./Jul. 2014.

[9] B. Li, J. Xu, X. Xu, S. Liu, and S. Lei, *CE2: Result of Test 1*, document JCTVC-S0080, Joint Collaborative Team on Video Coding, 19th Meeting, Strasbourg, France, Oct. 2014.

[10] C. Pang *et al.*, *Non-CE2: Intra Block Copy and Inter Signalling Unification*, document JCTVC-T0227, Joint Collaborative Team on Video Coding, 20th Meeting, Geneva, Switzerland, Feb. 2015.

[11] C. Lan, G. Shi, and F. Wu, "Compress compound images in H.264/MPGE-4 AVC by exploiting spatial correlation," *IEEE Trans. Image Process.*, vol. 19, no. 4, pp. 946–957, Apr. 2010.

[12] Z. Pan, H. Shen, Y. Lu, S. Li, and N. Yu, "A low-complexity screen compression scheme for interactive screen sharing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 6, pp. 949–960, Jun. 2013.

[13] X. Guo, Y. Lu, and S. Li, *RCE4 Test1: Major-Color-Based Screen Content Coding*, document JCTVC-P0108, Joint Collaborative Team on Video Coding, 16th Meeting, San Jose, CA, USA, Jan. 2014, pp. 9–17.

[14] L. Guo *et al.*, *RCE4: Results of Test 2 on Palette Mode for Screen Content Coding*, document JCTVC-P0198, Joint Collaborative Team on Video Coding, 16th Meeting, San Jose, CA, USA, Jan. 2014, pp. 9–17.

[15] Y.-C. Sun *et al.*, *AHG10: A Triplet Palette Mode Combining JCTVC-P0108 and JCTVC-P0198*, document JCTVC-Q0083, Joint Collaborative Team on Video Coding, 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[16] Y.-C. Sun *et al.*, "Palette mode—A new coding tool in screen content coding extensions of HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 2409–2413.

[17] P. Lai, S. Liu, T.-D. Chuang, Y.-W. Huang, and S. Lei, *Non-RCE4: Major Color Table (Palette) Sharing*, document JCTVC-P0153, Joint Collaborative Team on Video Coding, 16th Meeting, San Jose, CA, USA, Jan. 2014, pp. 9–17.

[18] Y.-C. Sun *et al.*, *SCCE3 Test A.2: Palette Coding and Palette Predictor Updates*, document JCTVC-R0119, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[19] C. Gisquet, G. Laroche, and P. Onno, *SCCE3: Test A.3—Palette Stuffing*, document JCTVC-R0348, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[20] B. Li, J. Xu, and G. J. Sullivan, *Comparison of Compression Performance of HEVC Screen Content Coding Extensions Test Model 5 With AVC High 4:4:4 Predictive Profile*, document JCTVC-V0033, Joint Collaborative Team on Video Coding, 22th Meeting, Geneva, Switzerland, Oct. 2015, pp. 15–21.

[21] Y.-C. Sun *et al.*, *SCCE3 Test B.4: Adaptive Color Index Map Scan*, document JCTVC-R0057, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[22] C.-M. Tsai, Y. He, X. Xiu, and Y. Ye, *SCCE3: Test B.9—BWT-Based Index Grouping*, document JCTVC-R0168, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[23] C. Betrisey *et al.*, "20.4: Displaced filtering for patterned displays," *SID Symp. Dig. Tech. Papers*, vol. 31, no. 1, pp. 296–299, May 2000.

[24] C. Gisquet, G. Laroche, and P. Onno, *Non-RCE4: Transition Copy Mode for Palette Mode*, document JCTVC-P0115, Joint Collaborative Team on Video Coding, 16th Meeting, San Jose, CA, USA, Jan. 2014, pp. 9–17.

[25] Y.-C. Sun, T.-D. Chuang, Y.-W. Chen, Y.-W. Huang, and S. Lei, *CE6 Test C.2: Transition Copy Mode*, document JCTVC-S0078, Joint Collaborative Team on Video Coding, 19th Meeting, Strasbourg, France, Oct. 2014.

[26] Y.-C. Sun, T.-D. Chuang, Y.-W. Chen, Y.-W. Huang, and S. Lei, *Non-RCE4: Cross-CU Major Color Index Prediction*, document JCTVC-P0093, Joint Collaborative Team on Video Coding, 16th Meeting, San Jose, CA, USA, Jan. 2014, pp. 9–17.

[27] Y.-C. Sun *et al.*, *Non-CE6: Cross-CU Palette Colour Index Prediction*, document JCTVC-S0079, Joint Collaborative Team on Video Coding, 19th Meeting, Strasbourg, France, Oct. 2014.

[28] Y.-C. Sun *et al.*, *CE1: Test A.1: Extended Copy Above Mode to the First Line (1.1-1.5)*, document JCTVC-T0036, Joint Collaborative Team on Video Coding, 20th Meeting, Geneva, Switzerland, Feb. 2015.

[29] [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.2+SCM-4.0/

[30] S.-T. Hsiang, T.-D. Chuang, and S. Lei, *AHG8: Coding the Prediction Differences of the Intra BC Vectors*, document JCTVC-Q0095, Joint Collaborative Team on Video Coding, 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[31] S.-T. Hsiang, T.-D. Chuang, and S. Lei, *SCCE1: Test 3.5–Block Vector Difference Coding*, document JCTVC-R0133, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[32] S.-T. Hsiang and S. Lei, *Unified BVD & MVD Coding Using a Universal Entropy Coding Scheme*, document JCTVC-R0134, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[33] S.-T. Hsiang, T.-D. Chuang, and S. Lei, *Non-SCCE3: Palette Index Coding Using a Universal Entropy Coding Scheme*, document JCTVC-R0135, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[34] S.-T. Hsiang, T.-D. Chuang, and S. Lei, *Non-SCCE3: Run Coding of the Palette Index Map Using a Universal Entropy Coding Scheme*, document JCTVC-R0136, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[35] S.-T. Hsiang, T.-D. Chuang, and S. Lei, *Non-SCCE4: Unified Method for Coding String Matching Syntax Using a Universal Entropy Coding Scheme*, document JCTVC-R0137, Joint Collaborative Team on Video Coding, 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[36] S.-T. Hsiang, T.-D. Chuang, and S. Lei, *CE1: Results of Tests 1.4, 2.2, & 3.3 on Unified BVD & MVD Coding*, document JCTVC-S0162, Joint Collaborative Team on Video Coding, 19th Meeting, Strasbourg, France, Oct. 2014, pp. 17–24.

[37] S.-T. Hsiang *et al.*, *CE6-Related: Harmonization of CE6 Tests A4, A5, and A6*, document JCTVC-S0269, Joint Collaborative Team on Video Coding, 19th Meeting, Strasbourg, France, Oct. 2014, pp. 17–24.

[38] S.-T. Hsiang and S. Lei, *CE2-Related: Improved Method for Entropy Coding MVD/BVD*, document JCTVC-T0127, Joint Collaborative Team on Video Coding, 20th Meeting, Geneva, Switzerland, Feb. 2015, pp. 10–18.

[39] S. H. Kim and A. Segall, *Non-CE2: Adding a Context for Significance Flag in BVD*, document JCTVC-T0114, Joint Collaborative Team on Video Coding, 20th Meeting, Geneva, Switzerland, Feb. 2015, pp. 10–18.

[40] X. Xu, T.-D. Chuang, S. Liu, and S. Lei, *Non-CE2: Intra BC Merge Mode With Default Candidates*, document JCTVC-S0123, Joint Collaborative Team on Video Coding, 19th Meeting, Strasbourg, France, Oct. 2014, pp. 17–24.

[41] X. Xu, T.-D. Chuang, S. Liu, and S. Lei, *CE2: Test 3.2–Intra BC Merge Mode With Default Candidates*, document JCTVC-T0073, Joint Collaborative Team on Video Coding, 20th Meeting, Geneva, Switzerland, Feb. 2015, pp. 10–18.

[42] M. Karczewicz, W. Pu, R. Joshi, and V. Seregin, *Non CE1: Grouping Palette Indices at Front*, document JCTVC-T0065, Joint Collaborative Team on Video Coding, 20th Meeting, Geneva, Switzerland, Feb. 2015, pp. 10–18.

[43] H. Yu, R. Cohen, K. Rapaka, and J. Xu, *Common Test Conditions for Screen Content Coding*, document JCTVC-T1015, Joint Collaborative Team on Video Coding, 20th Meeting, Geneva, Switzerland, Feb. 2015, pp. 10–18.

[44] G. Bjøntegaard, *Improvements of the BD-PSNR Model*, document VCEG-AI11, ITU-T SG16, Jul. 2008.

Yu-Chen Sun received the B.S. and M.S. degrees in electronic engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2004 and 2006, respectively, and the Ph.D. degree in computer science from National Chiao Tung University, Hsinchu, Taiwan, in 2013.

Since September 2013, he has been with MediaTek Inc. He actively contributes to digital video coding standardization developed by the ITU-T Study Group 16 (VCEG) and ISO/IEC JTC 1 SC 29 / WG 11 (MPEG) such as High Efficiency Video Coding (HEVC). He has authored or co-authored 28 standardization contribution documents, 13 journal/conference papers and 18 granted or pending patents in the field of video and image coding. His research interests include image/video processing, compression, and communication.

Shih-Ta Hsiang received the B.S. degree in electrical engineering from National Cheng-Kung University, Tainan, Taiwan, the M.S. degree in electrical engineering from University of Florida, Gainesville, FL, USA, and the M.S. and Ph.D. degrees in mathematics and in electrical, computer, and systems engineering, respectively, from Rensselaer Polytechnic Institute, Troy, NY, USA.

He was a Research Scientist with the Imaging Technology Department, Hewlett Packard Laboratories, Palo Alto, CA, USA, from 2002 to 2004, and a Principal Researcher with the Multimedia Research Lab, Motorola Labs, Schaumburg, IL, USA, from 2004 to 2011. He joined Corporate Technology Office, MediaTek Inc., Hsinchu, Taiwan, as a Senior Technical Manager in 2011, where he has been involved in research related to international video coding standardization activities.

Dr. Hsiang received the Allen B. DuMont Prize from the Rensselaer Polytechnic Institute in recognition of outstanding academic achievement.

Jungsun Kim received the Ph.D. degree in electrical engineering and computer science from Seoul National University, Seoul, South Korea.

In 2008, she joined LG Electronics as a senior research engineer and contributed to the Joint Collaborative Team on Video Coding. In 2015, she joined MediaTek, USA and has worked on video codec and its standardization. Her research interests includes video compression, video codec standardization and image processing.

Yi-Wen Chen received the B.S., M.S., and Ph.D. degrees in computer science and information engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2001, 2003, and 2011, respectively. He actively contributes to digital video coding standardization developed by ITU-T Study Group 16 (VCEG) and ISO/IEC JTC 1 SC 29 / WG 11 (MPEG) such as High Efficiency Video Coding (HEVC) and emerging 3D video coding standards such as MVC+D, 3D-AVC, MV-HEVC, and 3D-HEVC. He has authored or co-authored 150 standardization contribution documents, 30 journal/conference papers and 50 granted or pending patents in the field of video and image coding. His research interests include video/image compression, computer vision, video signal processing, content-based video indexing and retrieval, and multimedia information systems.

Xiaozhong Xu received the B.S. and Ph.D. degree from Tsinghua University, Beijing China, and the MS degree from Polytechnic school of engineering, New York University, NY, all in electrical engineering.

He is now with MediaTek USA Inc., San Jose, CA, USA, as a Senior Staff Engineer. Prior to joining MediaTek in June 2013, he worked for Zenverge, Inc., a semiconductor company working on multi-channel video transcoding ASIC design, from October 2011 to June 2013. He also held technical positions at Thomson Corporate Research (now Technicolor) and Mitsubishi Electric Research Laboratories. His research area lies in the general area of multimedia, including video and image coding, processing and transmission.

Dr. Xu has been an active participant in video coding standardization activities for over ten years. He has successfully contributed to various standards including H.264/AVC, AVS (China), HEVC and its extensions.

Weijia Zhu received the B.S. and Ph.D. degrees in computer science from the Beijing University of Technology (BJUT), Beijing, China.

He was a Deputy Technical Manager in MediaTek(MTK), Beijing, China. Now, he is a staff engineer in RealNetworks, Beijing, China. His current research includes video coding, image processing and computer vision.

ShawMin Lei (S’87–M’88–SM’95–F’06) received the B.S. and M.S. degrees from the National Taiwan University, Taipei, Taiwan, in 1980 and 1982, respectively, and the Ph.D. degree from the University of California, Los Angeles, CA, USA, in 1988, all in electrical engineering.

From August 1988 to October 1995, he was with Bellcore (Bell Communications Research), Red Bank, NJ, USA, where he had worked mostly in video compression and communication areas and for a short period of time in wireless communication areas. From October 1995 to March 2007, he was with Sharp Laboratories of America, Camas, WA, USA, where he was a manager of the Video Coding and Communication Group. Since March 2007, he has been with MediaTek, Hsinchu, Taiwan, as a Director of Multimedia Technology Division, working in the video/image coding/processing, computer vision, acoustics/speech processing, and bio-medical signal processing areas. His group has made significant contributions to the High-Efficient Video Coding (HEVC or H.265) standard. Under his direction, his group has become one of the top contributors in the video coding standard bodies, ISO MPEG and ITU-T VCEG. His research interests include video/image compression, processing and communication, picture quality enhancement, computer vision, and digital signal processing. He has published more than 90 peer-reviewed technical papers and more than 600 contributions to MPEG4, JPEG2000, H.263+, H.264, and HEVC international standard meetings. He has been awarded more than 100 patents.