Palette Mode Coding in HEVC Screen Content Coding Extension

Wei Pu, *Senior Member, IEEE*, Marta Karczewicz, Rajan Joshi, *Member, IEEE*, Vadim Seregin, Feng Zou, *Member, IEEE*, Joel Sole, *Senior Member, IEEE*, Yu-Chen Sun, Tzu-Der Chuang, Polin Lai, Shan Liu, *Senior Member, IEEE*, Shih-Ta Hsiang, Jing Ye, and Yu-Wen Huang

***Abstract*— Palette mode is a new coding tool included in the HEVC screen content coding extension (SCC) to improve the coding efficiency for screen contents such as computer generated video with substantial amount of text and graphics. It is observed that a local area in screen content typically has a few colors separated by sharp edges. To exploit such characteristics, palette mode represents samples in a block with indexes pointing to the color entries in a palette table. This paper provides a detailed overview of the palette mode in HEVC SCC in terms of palette generation, coding of the palette, and coding of the palette indexes for the samples in the palette block. Several improvements to palette mode coding, which have been proposed but not included in HEVC SCC, are also described. Simulation results are presented to quantify the bitrate savings provided by the palette mode for equal distortions.**

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

I. Introduction

THE first version of the High Efficiency Video Coding standard, also known as HEVC ver. 1 [1], demonstrated a substantial bitrate reduction over the H.264/AVC standard for camera captured natural video content. Although a great variety of test sequences with distinctive content characteristics were used during the standardization process, these camera captured sequences exhibit several common features such as edges smoothed due to diffuse reflections, white noise added due to the dust in the air, the imperfections of the lens and camera sensors, etc.

Recently, several new video applications have emerged. These include remote desktop sharing, virtual desktop infrastructure for cloud computing, WiFi wireless display, online gaming streaming, massive open online course (MOOC), and automotive display. These applications bring the need to compress, transmit, and display computer generated video content or a hybrid of computer generated and camera captured content (screen content). In contrast to natural video, screen content video sequences are usually rich in text and graphics, present sharp edges with less noise. They exhibit the property that the number of distinct colors within a local area in a picture is usually fairly small. Since the majority of the coding tools in HEVC ver. 1 targeted at camera captured video, they are not efficient in exploiting and removing the redundancies in screen contents.

To promote these new video applications, in January 2014, the ISO/IEC JTC 1/SC 29/WG 11 (also known as Moving Picture Experts Group, MPEG) Requirement subgroup published a set of requirements for a screen content coding extension of the HEVC standard (referred to as HEVC SCC hereafter), targeting on improving coding efficiency for three types of screen content: mixed content, text and graphics with motion, and computer animation. Given these requirements, ITU-T Study Group 16 the Video Coding Experts Group (VCEG) and the Moving Picture Experts Group (MPEG) issued a joint Call for Proposals (CfP) [2]. Seven responses to the CfP were received during the 17th JCT-VC meeting in Valencia, Spain in March 2014. These responses formed a basis for the development of the HEVC SCC. The HEVC SCC reached the final draft international standard (FDIS) stage in February 2016. This paper presents a detailed overview of a new major coding tool named palette mode in HEVC SCC, which exploits the special property of exhibiting only a few colors separated by sharp edges in screen content. For a general high level overview of the screen content coding extension including other tools such as adaptive color transform and intra block copy, please refer to [3]. It is worth noting that although palette has been successfully used for over two decades in the popular Portable Network Graphics image format (PNG) and Graphics Interchange Format image format (GIF), the palette mode in SCC introduced a substantial amount of new algorithms to improve palette coding efficiency, such as palette predictive coding, context based run length coding, variable length index coding, which are not observed in PNG or GIF.

This paper is organized as follows. Section II presents the technical principles and coding methods for the palette mode. Section III introduces several promising ideas for the improvement of the palette mode proposed during the standardization process. The purpose of presenting these ideas is to identify opportunities for further improvements to the palette

Manuscript received February 11, 2016; revised May 23, 2016 and July 22, 2016; accepted August 5, 2016. Date of publication November 18, 2016; date of current version December 9, 2016. This paper was recommended by Guest Editor R. Cohen.

W. Pu is with Qualcomm Inc., Uber Advanced Technologies Center, Pittsburgh, PA 15201 USA (e-mail: wpu.multimedia@gmail.com).

M. Karczewicz, R. Joshi, V. Seregin, and J. Sole are with Qualcomm Technologies Inc., San Diego, CA 92121 USA (e-mail: martak@qti.qualcomm.com; rajanj.qualcomm.com; vseregin@qti.qualcomm.com; joels@qti.qualcomm.com).

F. Zou is with Hulu, LLC, Los Angeles, CA 90404 USA (e-mail: fenzou@gmail.com).

Y.-C. Sun, T.-D. Chuang, P. Lai, S. Liu, S.-T. Hsiang, J. Ye, and Y.-W. Huang are with MediaTek Inc., Hsinchu 30078, Taiwan (e-mail: ckey.sun@gmail.com; Peter.Chuang@mediatek.com; Polin.Lai@mediatek.com; Shan.Liu@mediatek.com; shih-ta.hsiang@mediatek.com; Jing.Ye@mediatek.com; yuwen.huang@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.2016.2605661

mode and to spur further research related to the palette mode. Section IV presents simulation results and Section V concludes the paper with a summary.

II. Palette Mode: A New Coding Tool in HEVC SCC

In screen content video, it is observed that within a coding unit (CU, the basic processing unit composed of one luma block and ordinarily two collocated chroma blocks [1]), the number of distinct colors is usually fairly small. This is the motivation for the introduction of a palette mode in HEVC SCC, which represents pixel samples in a CU with sample indexes pointing to a finite set of colors (the palette). Palette mode has been investigated for years after the success of PNG and GIF [4], [5]. It is widely known in video coding area that palette, as a new coding mode, can be integrated into the block based hybrid coding architecture to improve the coding efficiency for screen content. However, how to design the specific coding algorithms, e.g. palette coding and index coding, to achieve the most competitive coding gain with reasonable hardware and software implementation complexity is a serious research and engineering challenge. During HEVC SCC standardization, several palette mode coding methods were proposed [6]–[8], trying to answer such challenge.

A typical palette mode coding method is composed of two parts: coding methods for the palette and coding methods for the samples using the palette. The latter part is composed of palette index coding, run length coding, and escape pixel coding. In [6], each color component of a CU has its own palette, and each sample of a color component is coded with one index. In [7], the three color components share a single palette where each palette entry consists of three components. Each Y (G) sample and its corresponding C_b (B) and C_r (R) samples are grouped and coded with a single index. In [8], a combination of individual component palettes and a combined component palette is introduced. In [6], palette indexes are coded sample by sample with a different choice of sample prediction modes, or an entire sample line in the CU is coded with one of two line modes. One line mode copies the previous line entirely, while the other line mode duplicates the first sample for the entire line. In [7], indexes are coded sample by sample with or without prediction, or multiple samples are coded with a run mode that allows duplicating indexes across sample lines. Based on [6], an improved candidate-based predictive coding method is proposed in [8], where the candidate set consists of the left, top, top-left, and top-right neighboring samples. With pruning out redundant candidates, good coding gain can be achieved. To exploit the advantages from multiple proposals, a combined palette mode was later proposed [9]. Compared with the previous individual proposals, the combined palette mode showed significantly higher coding efficiency and lower complexity. The combined palette mode was adopted into the HEVC SCC under development in the 18th JCT-VC meeting in Sapporo Japan in July 2014. In subsequent meetings, palette improvements were studied on top of [9]. The following subsections describe the details of the palette mode in the HEVC SCC and some important non-normative techniques.

TABLE I
Toy Example of a Palette

Index	Color Intensity (R, G, B)
0	2, 10, 200
1	0, 0, 0
2	10, 10, 180
3	10, 20, 72

0, 10, 200	0, 128, 200	0, 0, 0
2, 10, 200	120, 110, 200	0, 10, 200
4, 10, 200	10, 10, 180	10, 20, 70

Classification

Index 0	Escape	Index 1
Index 0	Escape	Index 0
Index 0	Index 2	Index 3

Fig. 1. An example of pixel classification in palette mode, using the palette example in Table I. Above matrix: each individual pixel's RGB color triplet. Below matrix: each individual pixel's class.

A. Basic Elements: Palette, Index, and Escape Samples

A Palette mode is signalled at the CU level and supported for CUs of size less than or equal to 32×32 [10]. For a CU coded in the palette mode, a palette, which enumerates the dominant colors within the CU, is signalled in the bitstream. The palette is generally implemented as a color lookup table in which each color entry is associated with an index. Table I is a toy example of a palette with four entries, indexed from 0 to 3. Each entry represents a specific RGB color. For example, entry 1 with (R, G, B) = (0, 0, 0) represents pure black color while entry 0 with (R, G, B) = (2, 10, 200) represents a bluish color.

Once the palette is constructed, samples within the CU can be classified into two categories. A sample belonging to the first category is the same or very close to an entry in the palette. In this case, the sample can be represented by the index of its corresponding entry in the palette. It is up to the encoder to define a distortion metric and a threshold to determine whether a sample is close enough to a palette entry. Typical distortion metrics used are sum of absolute differences (SAD) and sum of squared errors (SSE). The decoder can reconstruct the sample by looking up the palette entry using the corresponding index. These samples are referred to as *indexed samples*. For samples belonging to the other category, each sample is significantly different from any entry in the palette. These samples are not suitable to be represented by a palette index and are referred to as *escape samples.* Their color component values are quantized and explicitly coded in the bitstream. Fig. 1 is an example of classifying a 3×3 color block using the palette in Table I.

The following subsections present the key procedures for designing and implementing high efficiency palette mode, as well the extension to non 4:4:4 chroma subsampling format.

B. Palette Generation

The generation of palette is non-normative but can be vital to the performance of the palette mode. One straightforward way to design a palette is to choose the first few dominant colors in the CU. Since typically, an encoder has the flexibility of choosing any value as a palette entry, picking up the first few dominant colors may not be optimal. Furthermore, the palette generation is also influenced by the method used to code the palette entries. For example, as will be described in more detail in Section II-C, palette entries from the neighboring palette mode coded CUs are cached in a buffer, and used to predict the palette entries of the current palette. This buffer is named as palette predictor list. It influences the generation of the palette as well. Some of the aspects considered in the generation of the palette are as follows.

1) The frequency of a color in the CU: The more times a color appears in the CU, the more dominant it is.
2) The distortion threshold: If a higher distortion is allowed, similar colors may be grouped together as a single entry in the palette. Although each individual color in the group may occur only a few times, the combined frequency may be significant.
3) Whether a palette entry can be predicted: If an entry being considered for inclusion in the palette is present in the palette predictor list, the overhead for coding that entry in the bitstream is very small, leading to a more favorable rate-distortion trade-off.
4) An upper limit on the palette size based on the profile and level: This number is signalled in the sequence parameter set (SPS) and is denoted as *palette_max_size* in the HEVC SCC draft specification [11]. A small limit reduces the number of bits needed to signal the palette and the palette indexes for the samples in the block. It also reduces hardware implementation costs. However, a small limit may introduce a higher cost in coding the samples in the CU due to higher distortion or more samples classified as escape samples. This represents a design trade-off.

Many methods were proposed during the standardization process to jointly consider these aspects [12]–[14]. In the latest SCC reference software SCM5.2 [15], a two-pass encoding is used to derive the palette, which is described as the following.

For lossy coding, in the first pass, a palette is generated mainly based on the histogram of sample values in the CU. In the second pass, a competing candidate palette is generated with bias towards palette entries that are present in the palette predictor list. A CU is coded twice with respect to the two candidate palettes respectively. The one generating smaller rate-distortion cost is used as the actual palette for the CU.

When calculating the dominant color, similar colors are grouped together if their distances are within a palette quantization step *pltQStep*. This value is derived according to the following formula:

$$pltQStep = \left\lfloor \frac{2 \times QStep}{3} + 0.5 \right\rfloor \tag{1}$$

where *QStep* is the quantization step-size for the current CU. This formula was developed by training a linear regression model with *QStep* as input and *pltQStep* as output, targeting on minimizing rate-distortion cost.

The centroid of the samples within each group, after rounding to the nearest integer, is used to represent the group. For each centroid, a rate-distortion comparison is used to decide whether to put it into the palette of the current CU or to re-use one entry from the palette predictor list.

On the other hand as for lossless coding, in the first pass, all the colors in the CU are sorted according to their frequencies of occurrence. The leading high frequency entries (up to *palette_max_size*) compose the initial palette for the current CU. A refinement procedure is used for the entries which occur only once in the CU. Such entries are kept in the palette only if they are in the palette predictor list. In the second pass, a competing palette is generated by selecting entries in the palette predictor list as long as they reoccur at least once in the current block. If the number of such entries is greater than *palette_max_size*, the encoder picks the first *palette_max_size* entries according to their frequencies. Then, a procedure similar to the first pass is invoked to generate the rest of the palette entries based on the statistics of samples that are not covered by the predicted palette entries.

There are some further improvements on palette generation. Readers may refer to [12], [13] for the details.

C. Coding of Palette Entries

It is necessary to code information about the current palette in the bitstream so that the decoder can reconstruct the same palette as used by the encoder. A straightforward way is to code each entry in the palette using a fixed length code (FLC). For a typical video sequence having a bit depth of 8 bits/component, 24 bits are needed to code a palette entry using FLC. To improve coding efficiency, predictive coding methods were developed, based on the observation that neighboring palette mode CUs may share several palette entries with the current palette.

1) Palette Predictor List:: In HEVC SCC, a palette predictor list stores the previously coded palette entries as references to predict the current palette [8], [16]–[18]. This list is updated after each palette mode CU. It operates like a least recently used cache. The latest palette is inserted at the beginning of the list and the entries from the farthest CUs in scan order are discarded if the list size exceeds a threshold. The upper bound on the size of the palette predictor list is (indirectly) signalled in the SPS header. In a typical configuration, it is chosen to be roughly twice the size of the palette size limit [19]. Some examples can be found in [20].

It is observed that slices and tiles can result in substantial coding efficiency loss as the palette predictor list is assumed to reset at the slice and tile boundaries. After the reset, it takes time to rebuild the palette predictor list. A palette initializer with predefined entries can be optionally used to initialize the palette predictor list, resulting in improved coding efficiency in such a scenario [21]. A palette initializer can be signalled in the picture parameter set (PPS) or the sequence parameter set (SPS).

When wavefront processing is used, the palette predictor list is initialized in a similar way as CABAC contexts. In each

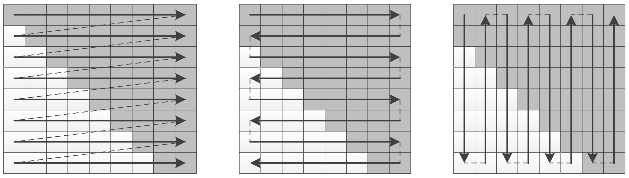


Fig. 2. Sample scanning order (from left to right): raster horizontal scanning (not used in palette mode), horizontal traverse scanning, and vertical traverse scanning.

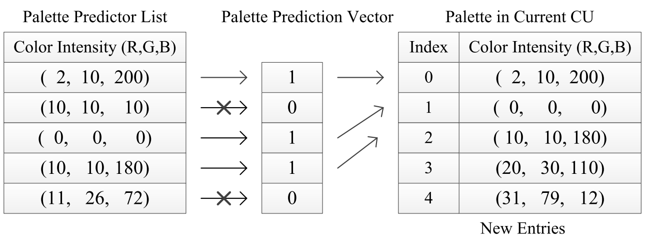


Fig. 3. Example of palette prediction vector.

coding tree unit (CTU) row, the palette predictor list is stored after encoding the second CTU in the row, and the stored palette predictor list is used for initialization of the next CTU row [22], [23].

2) Palette Prediction Vector:: To code the current palette using the palette predictor list as a reference, a binary vector is used to indicate whether each entry in the palette predictor list is reused in the current palette. The reused entries are placed at the beginning of the current palette, maintaining their order in the palette predictor list. This is followed by new palette entries which are not in the palette predictor list.

Fig. 3 shows an example of a palette predictor list and a palette prediction vector. There are five entries in the palette predictor list before coding the current CU. First, a binary vector "10110" is signalled, indicating that the first, third, and fourth elements in the palette predictor list are re-used in the current palette. These three elements are placed at the beginning of the current palette, indexed as 0, 1, and 2, respectively. In the example, the current palette has two more entries (index 3 and 4) which are not predicted from the palette predictor list. The component values of these two entries are transmitted directly using a fixed length code.

In an empirical configuration used as the SCC common test conditions [24] which demonstrated good coding performance, the maximum palette size is chosen to be 64 and the maximum palette predictor list size is chosen to be 128. Run length coding [25] is used to compress the binary palette prediction vector. The run length based palette prediction vector coding is developed based on the observation that in the binary vector, there are long substrings of zeros, while substrings of ones are usually short. Therefore, a run length coding scheme for substrings of zeros is developed. Exponential Golomb (exp-Golomb) code of order 0 (EG0) is used to code the run length. It is inferred that a run of zeros is followed by a 1 unless the end of the vector is reached. As a further optimization, the EG0 codeword corresponding to a run length of one is

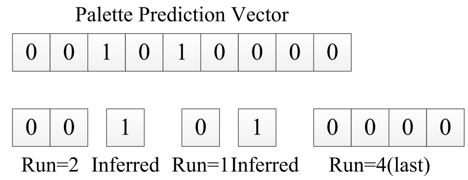


Fig. 4. Example of run length based palette prediction vector coding.

reserved to indicate that there is no more 1 in the vector. After decoding a run length of one, all the bins from the current position to the end of the vector are inferred to be zeros. If the decoder decodes a run length greater than one, the run length is adjusted downwards by one to account for this.

Fig. 4 is an example of palette prediction vector coding. In the example, the run length sequence to be coded using EG0 is 2, 1, 4. Since the run length of one is reserved for the last run, the run lengths greater than zero are incremented by 1 before coding. Thus, the coded run lengths in the example are 3, 2, 1.

D. Coding of the Sample Indices

1) Scanning Order:: To code the samples in the CU, two different scanning orders, horizontal traverse scanning and vertical traverse scanning, may be used. A CU level flag is coded in the bitstream to indicate the scanning direction. Traverse scanning eliminates the spatial discontinuity in raster scans at the end of rows (or columns for vertical scans) [26]. This results in longer runs when using run length coding to code the palette indexes as described in Section II-D3). Horizontal and vertical scanning directions are well-suited for text contents written column-by-column and row-by-row, respectively [27]. Fig. 2 illustrates the sample scanning orders in the palette mode. In the following sections, unless explicitly stated, horizontal traverse scanning is assumed.

2) Pixel Marking and Coding:: As briefly mentioned in Section II-A, for decoding of escape samples, quantized component values need to be parsed. Hence the decoder needs to know the category for each sample, i.e., whether it is an indexed sample or an escape sample. Two alternative approaches were developed during the HEVC SCC standardization process to code such information.

In the first approach, for a given sample, one CABAC context coded bin is signalled to indicate whether it is an escape sample [28]. In an alternative approach, a virtual index equal to the palette size is reserved to represent an escape sample [6]. If the decoded index is equal to the palette size, the sample is an escape sample. The first method provides small coding gain at the cost of increasing the number of context coded bins per sample (a metric commonly used to estimate the CABAC engine throughput) by one in the worst case. The second method streamlines index processing since every sample in the block is assigned a palette index. Due to these two considerations, the JCT-VC committee decided to adopt the second method for representing escape samples.

To code a palette index, truncated binary code (TBC) [29] is used. TBC is an extension of fixed length coding (FLC).

TABLE II
EXAMPLE OF A TRUNCATED BINARY CODE ($N = 6$)

x	FLC	TBC
0	000	00
1	001	01
2	010	100
3	100	101
4	101	110
5	110	111

To code a uniformly distributed symbol $x \in \{0, 1, \ldots, N-1\}$, FLC is not optimal when $2^k < N < 2^{k+1}$, as FLC uses $k+1$ bits per symbol while the information theoretical minimum average codeword length bound is $\log_2 N$, which is less than $k+1$. TBC improves FLC by using codewords with lengths k or $k+1$ depending on the value to be coded to achieve a shorter average codeword length. It is worthy to note that the uniform distribution assumption of the coded indexes does not contradict the skewed overall index distribution assumption. Using the run length coding scheme which will be discussed in the next subsection, only a small portion of the pixel samples need to code their indexes into the bitstream. Especially, because the high frequency indexes tend to have longer run and we need to code at most one index per run, the actually coded number of indexes for high frequency indexes is much smaller than their actual occurrences in the block. So the actually coded indexes tend to be uniformly distributed. This phenomenon is also observed practically from simulation. Table II is an example of a TBC for $N = 6$. In this example, on average, TBC requires $2 \times \frac{2}{6} + 3 \times \frac{4}{6} = 2.67$ bits per symbol while FLC requires $\lceil \log_2 N \rceil = 3$ bits per symbol. As a reference, the theoretical lower bound is $\log_2 N = 2.58$ bits per symbol. We can see that for small values of N, use of TBC results in a substantial coding gain.

When an index is reserved for the escape sample, the dynamic range of the indices is increased by one. As a result, the average bit cost per coded index increases to $\log_2(N+1)$. It is observed that only a portion of the palette CUs contain escape pixels, especially for high QPs (i.e., when the bit budget is low). To exploit this property, a CU level flag is signalled to indicate whether the current CU contains at least one escape sample. If the flag is 0, the dynamic range of the indices is kept unchanged at $[0, N)$.

If a sample is an escape sample, after signalling its index, quantized component values are coded in the bitstream. It was initially proposed to use TBC to code these values [30]. However, this approach requires additional calculation of the maximal possible quantized value depending on the block's QP and component bit depth. Hence the JCT-VC committee decided to use exponential Golomb code of order 3 (independent of the QP) to code the quantized component values for escape samples [31]. In lossless coding, escape samples are not quantized and component values are signalled directly using a binary representation based on the component bit depth.

3) Run Length Coding:: Each sample in the palette coded CU is associated with an index. Run length coding is used to compress the two-dimensional (2D) index array.

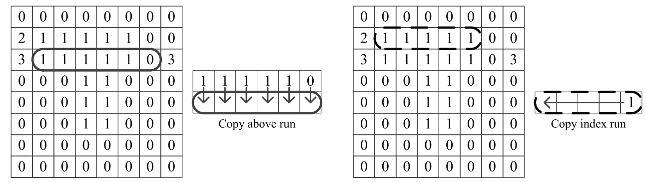


Fig. 5. Run length coding of the 2D index array. Left: "copy above" run. Right: "copy index" run.

Two types of runs are used, as shown in Fig. 5. For the first type, referred to as the "copy index" run, an index value is signalled in the bitstream followed by a run length value indicating the number of subsequent samples in the scanning order that share the same index value as the signalled one. In the example in Fig. 5, an index value equal to 1 is signalled first. This is followed by a run length value of 4, indicating that the index value for the four subsequent samples (from right to left) is also equal to 1.

For the second run type, referred to as the "copy above" run, no index signalling is necessary. Only a run length value is coded indicating the number of samples following the current sample that share the same index as their respective above (or left for vertical traverse scan) adjacent neighbors. Referring to Fig. 5, a "copy above" run length value of 5 is used to indicate that the current sample as well as the five subsequent samples in the scanning order (six samples in total) has the same index value as their respective above adjacent neighboring samples. A flag is transmitted to indicate the run type for each run. However, if a "copy above" run is chosen by the previous group of samples, the current group of samples is normatively forbidden from choosing the "copy above" run. For example, a group of M pixels coded with a "copy above" run followed by a group of N pixels coded with a "copy above" run is essentially equivalent to a group of $(M+N)$ pixels coded with a "copy above" run. Thus, the run type selection flag is not transmitted for the second group and the second run is inferred to be a "copy index" run [32]. On the encoder side, if a sample can be coded either in a "copy index" run or a "copy above" run, a rate-distortion comparison is used to make the decision [33].

The entropy coding method, first proposed in [34], is employed for coding run length values. This method represents an unsigned source sample value in a binary form by first signaling the position of its most significant bit (MSB) index followed by its refinement bits. An MSB index of -1 is assigned for a sample value equal to 0. The prefix part of the resulting codeword represents the MSB index plus 1, denoted by *msb_id_plus1*. Given an unsigned source sample value x, *msb_id_plus1* is defined as

$$msb\_id\_plus1 = \begin{cases} floor\left(log_2\left(x\right)\right) + 1, & x > 0 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where $log_2(\cdot)$ represents the base-2 logarithm and $floor(\cdot)$ represents 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*

TABLE III
CONTEXT ASSIGNMENT FOR RUN LENGTH

Bin	Copy above	Copy index
0	0	3, 4, 5
1	1	6
2	1	6
3	2	7
4	2	7

is greater than 1. Given an unsigned source sample value x, *refinement_bits* is defined as

$$refinement\_bits = x\&((1 \ll (msb\_id\_plus1 - 1)) - 1) \tag{3}$$

where "&" represents the bit-wise AND operator, and "≪" represents left bit shift. The decoded sample value x can be derived as

$$x = \begin{cases} msb\_id\_plus1, & msb\_id\_plus1 < 2 \\ (1 \ll (msb\_id\_plus1 - 1)) & \\ \quad + refinement\_bits, & \text{otherwise} \end{cases} \tag{4}$$

For application to palette run length coding, this method is employed for coding the run length value of each palette run [35]–[37]. A truncated binarization is used with the maximum code value equal to the maximum possible run length for the current palette run. The prefix part *msb_id_plus1* is binarized by a truncated unary code and context-coded using CABAC (up to the first five bins). The suffix part *refinement_bits* is binarized by a truncated binary code and entropy coded in CABAC bypass mode. The resulting binarization is equivalent to the truncated version of the exp-Golomb code proposed in [38].

The context models for entropy coding the binarized prefix *msb_id_plus1* are determined by the location of the bin in the binarization string, the index value, and the palette run type [35]–[37], [39], [40]. Table III is the context assignment table for *msb_id_plus1* in HEVC SCC. First, it is observed that the statistical distributions of run lengths in "copy above" runs and "copy index" runs are very different. Therefore, two sets of CABAC contexts, based on the run type, are used. Second, it is also observed that the histogram of palette indices is much skewed towards smaller indices. This is partially due to the fact that for a typical encoder implementation, such as the SCM5.2 reference software, frequently used entries are generally placed at the beginning of the palette to improve coding efficiency. Therefore, contexts are split according to the location of the bin, which reflects the impact of run length information. In the third dimension, for the "copy index" run type, the run lengths tend to be longer for smaller index values. To exploit this feature, context modeling for the first bin is further conditioned on the corresponding palette index. Specifically, when the corresponding palette index is equal to 0, context 3 is used. If the index equals to 1 or 2, context 4 is used. Otherwise, context 5 is used. The parsed palette index (to be explained in the next section) is employed for context selection to avoid bitstream parsing dependency. Also as shown in Table III, some bins (e.g., bins 1 and 2) share the same context to reduce the total number of contexts. The bins corresponding to bin index values greater than 4 are entropy coded in a bypass mode. In this way, in the worst case, no more than 5 context coded bins are used to code each palette run length.

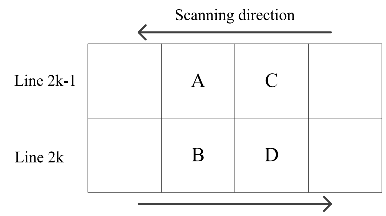


Fig. 6. Example of index dynamic range reduction. D is the starting pixel in the current copy index run.

4) Index Redundancy Removal:: In the palette mode, some normative restrictions are imposed to make sure that the length of the run is maximized. For example, referring to Fig. 6, let sample D be the current position, and also the start of a "copy index" run. Let the index value for sample D be denoted by index_D. If the previous run is a "copy index" run, it is normatively required that index_D is not equal to index_B. This is because if index_D is equal to index_B, the two consecutive "copy index" runs could be merged into a single run. Similarly, if the previous run is a "copy above" run, there is a normative restriction that index_D is not equal to index_C. Otherwise, the previous "copy above" run could be extended by one. In HEVC-SCC, these normative restrictions are exploited to reduce the dynamic range of the index to be coded by one, except for the first sample in the block. When the palette size is small, reducing dynamic range by one can result in substantial coding gain, as reported in [43].

Referring to Fig. 6, let the dynamic range of index_D be N, i.e. the index value is chosen from $\{0, 1, \ldots, N - 1\}$. Specifically, if there is no escape sample in the CU, N is equal to the palette size. If the palette CU contains escape samples, N is equal to palette size plus one. If the previous pixel in scan order, i.e. B, is the end of a "copy index" run, index_D must not be equal to index_B. To code index_D, if $\text{index}_\text{D} > \text{index}_\text{B}$, a value of $\text{index}_\text{D} - 1$ is coded by TBC with dynamic range adjusted to $N - 1$. Otherwise, when $\text{index}_\text{D} < \text{index}_\text{B}$, a value of index_D is coded by TBC with range adjusted to $N - 1$. On the decoder side, if the decoded symbol value S is greater than or equal to index_B, then index_D is set to equal to $S + 1$. Otherwise, index_D is set equal to S. Similarly, if B is the end of a "copy above" run, similar dynamic range reduction procedure can be applied using index_C instead of index_B.

Numerically, using TBC, the average bits per index is approximately equal to $\log_2(N)$. After the dynamic range adjustment, the average bits per index is approximately equal to $\log_2(N - 1)$. The percentage of bit savings due to the dynamic range adjustment is equal to

$$\frac{\log_2(N) - \log_2(N - 1)}{\log_2(N)} \tag{5}$$

Fig. 7 visualizes such savings for different values of N. It shows that when N is small, the improvement is substantial.

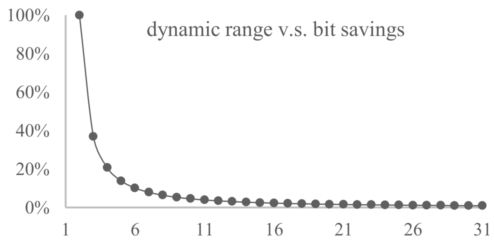


Fig. 7. Bit saving (percentage) due to dynamic range reduction versus the original index dynamic range.

During the standardization process, a concern was raised that when such dynamic range adjustment is applied, use of the adjusted index value to determine the context for run length coding would introduce parsing dependency. In other words, after parsing the index value, it would be necessary to perform index adjustment before parsing the run length. To resolve this issue, the parsed index (i.e., before index adjustment) is used to determine the context used in run length coding.

E. Grouping of Indices and Escape Pixel Values

The CABAC engine operates in two different modes, namely bypass mode and normal mode [44]. In the normal mode, a probability model (context) is derived based on previously coded binary symbols (bins). After coding the input bin, the associated probability model is updated. In the bypass mode, an equal probable model is assumed. In this mode, there is no context derivation and model update. One of the guiding principles in the design of HEVC entropy coding is grouping bypass bins to increase throughput [45].

In the initial palette mode design, the syntax elements of palette run type (i.e., the flag specifying "copy above" run or "copy index" run), index value, run length, and quantized escape sample values were interleaved. This means normal bins and bypass bins are interleaved. For example, palette run type is in normal mode. Palette index is coded in bypass mode. Run length is coded with a combination of normal and bypass mode. Finally, the quantized escape samples are coded in bypass mode. Such syntax element layout is not implementation friendly.

In [46], it was proposed to group together the bypass coded bins for coding the palette indices to improve CABAC throughput. Similarly, [47] and [48] proposed grouping together the bypass coded bins corresponding to quantized escape samples at the end of the palette coded block. Finally, in order to reduce coefficient buffers of the decoder, [49] and [50] proposed that the escape samples of the same color component are grouped together first and then signalled group by group. These changes were adopted by the JCT-VC committee and are a part of the HEVC SCC specification. As a summary, the palette mode CU syntaxes can be roughly partitioned into four groups as follows.

1) Syntax elements related to coding of the palette.
2) Syntax elements related to palette indices corresponding to the "copy index" run type.
3) Palette run type and run lengths.
4) Quantized escape samples for each color component.

All the context-coded bins are primarily concentrated in group 3.

When grouping indices at the beginning, a new syntax element representing the total number indices in the block is coded in bypass mode. This is followed by individual indices. It is observed that for palette coded blocks, due to the character of screen content, the last run in the block is typically quite long [41]. Based on this observation, a flag is inserted into the bitstream after the coding of the indices, specifying whether the last run type is "copy index" or "copy above". This enables the encoder to skip coding of the run length corresponding to the last run. On the decoder side, when the decoder has used up all the decoded indices in the CU, if the next run type is the same as the last run type flag, it is inferred to be the last run and the decoding of the last run length is skipped.

F. Non-4:4:4 Chroma Subsampling Support

To extend the palette mode to non-4:4:4 chroma formats, a single palette is used to code both luma and chroma components [52], [53]. The syntax is kept almost identical to the 4:4:4 chroma format case. The only exception is that if a sample is an escape sample and the sample position only contains a luma component, only one quantized component is included in the bitstream instead of three. Similarly, for the reconstruction of non-escape samples, one or three components from the palette entry corresponding to the palette index are used, depending on the position of the sample. The benefit of this approach is that it aligns the palette mode for non-4:4:4 chroma formats with the 4:4:4 chroma format. This results in a simpler design at the cost of some coding efficiency.

III. Extensions

During the standardization stage, several additional ways to improve the coding efficiency of the palette mode were proposed. These were not adopted due to a variety of reasons such as rate-distortion performance versus implementation complexity trade-off, timing and maturity of the proposals, etc. In this section, three representative techniques are presented which may point to future research directions for improving the palette mode.

A. Copy Previous

The success of "copy above" run type in improving coding efficiency can be extended by allowing copying of indices from rows that were not immediately above, but further away.

In [54], an additional run type, "copy previous," was proposed which is a generalization of the "copy above" run type in HEVC SCC. A syntax element k, which specifies the offset to the row from which indices are copied, is coded into the bitstream. When k equals to 0, the "copy previous" mode is the same as the normal "copy above" mode. When k is greater than 0, the redundancy between the current line and k lines above is exploited. This method achieves over 1% coding efficiency gain, as reported in the original proposal. The main concern about this method was that to achieve such gain, the encoder may need to exhaustively search every candidate line above the current line, which is not desirable.

B. Copy Outside

The two run types currently included in the HEVC SCC specification, i.e., "copy above" and "copy index," exploit local sample correlations among adjacent neighbors. But for the first row of samples, "copy above" run type is not possible and hence, "copy index" mode is inferred. In [55] and [56], it is proposed to allow the "copy above" run type in the first row of the CU. If "copy above" mode is chosen in the first row of the CU, reconstructed neighboring samples outside the current palette block are copied. The approach effectively brings information from neighboring blocks into the current block. Two alternative approaches were proposed. In the first one [55], a neighboring sample is first assigned a palette index whose corresponding value is the closest to it. However, the sample to index conversion operation introduces additional complexity, especially on the decoder side, which is undesirable. In the other approach [56], neighboring sample values are directly used without converting them to palette indexes. This approach saves the computation of sample to index conversion. However, in this approach, certain samples inside the palette block do not have a valid index if they are copied from neighboring samples outside the current block. As the index dynamic range adjustment operations described in Section II-D4) implicitly assumes that each pixel in the block has a valid index, the process of dynamic range adjustment has to be modified to harmonize these two methods [57], [58].

Finally, it is worthy to point out that in [59], it is reported that copy previous and copy outside can be used in combination to achieve over 3% additive gain.

C. Dual Palette Solution for Non-4:4:4 Format

Different from the single palette approach in HEVC SCC, in [51], a dual palette approach was proposed by coding luma component and chroma components independently. One palette is used for the luma component and another palette is used for the chroma components. As the chroma component block is more homogeneous and has less number of samples compared with the luma component block for non-4:4:4 chroma formats, this method achieves significantly higher coding efficiency.

Since the focus of the HEVC SCC development was the 4:4:4 chroma format, the JCT-VC committee decided that the simpler single palette approach discussed in the previous section of aligning the palette mode for all chroma formats was preferable and was adopted into the standard.

D. Termination Flag for Run Length Coding

Further improvements to run length coding of palette indexes were proposed in [41], [42]. In [41], it was proposed to insert a flag between the prefix and suffix of the truncated exp-Golomb code when the prefix unary code is truncated, to indicate whether the current run reaches the end of the block. If the flag is 1, the suffix bits are bypassed. In [42], it is proposed to use a flag to indicate whether the current run spans multiple of whole lines when the starting position is the first sample in a line in scanning order. If the flag is equal to 1, instead of coding the actual run length, the number of lines spanned by the run is coded. These improvements were not adopted into the HEVC SCC, since the JCT-VC committee decided that for these methods, the trade-off between rate-distortion improvement and implementation complexity was not favorable.

There are some other interesting ideas proposed during the development of HEVC SCC, such as transition copy [60], [61] and combination of palette and intra line copy [62]. It is worth noting that the coding gains of transition copy and copy previous are additive [63]. Due to space constraints, these methods are not included in this paper. Readers may refer to the original proposals for details.

TABLE IV
TEST SEQUENCES IN SCC CTC

Resolution	Sequence name	Cat.
1920×1080	sc_flyingGraphics_1920x1080_60_8bit	TGM
	sc_desktop_1920x1080_60_8bit	TGM
	sc_console_1920x1080_60_8bit	TGM
	ChineseEditing_1920x1080_60_8bit	TGM
	MissionControlClip3_1920x1080_60p_8b444	M
	EBURainFruits_1920x1080_50_10bit	CC
	Kimono1_1920x1080_24_10bit	CC
1280×720	sc_web_browsing_1280x720_30_8bit	TGM
	sc_map_1280x720_60_8bit	TGM
	sc_programming_1280x720_60_8bit	TGM
	sc_SlideShow_1280x720_20_8bit	TGM
	sc_robot_1280x720_30_8bit	A
2560×1440	Basketball_Screen_2560x1440_60p_8b444	M
	MissionControlClip2_2560x1440_60p_8444	M

IV. SIMULATION RESULTS

This section presents simulation results using the HEVC SCM 5.2 reference software [15] under the SCC common test conditions [24].

A. Test Conditions

A set of common test conditions (CTC) were used to evaluate the performances of new algorithms during the development of HEVC SCC. CTC define three prediction categories, i.e., all intra (AI), random access (RA), and low delay B (LDB). For each category, for lossy coding, four QPs{22, 27, 32, 37} are used to report the BD-rate results [64].

There are 14 test sequences in the CTC, with a variety of different resolutions and content characteristics. Table IV is a summary of the test sequences. "TGM," "CC," "M," and "A" stand for Text and Graphics with Motion, Camera Captured, Mixed content (i.e. content as a mixture of TGM and CC), and Animation, respectively. For each sequence, two chroma formats (RGB and $YC_{\{b\}}C_{\{r\}}$, denoted as YUV in the tables) are tested. For more details on the CTC, readers may refer to [24]. In this section, only RGB444 and YUV444 format results are reported since they were the main focus of the HEVC SCC developments.

TABLE V
BD-RATE CHANGES FOR SCM5.2 WITH PALETTE VERSUS WITHOUT PALETTE (INTRA BLOCK COPY IS DISABLED)

Cat.		AI	RA	LD-B
RGB	TGM	-39.5%	-28.8%	-17.5%
	M	-19.8%	-14.0%	-7.6%
	A	-0.2%	-0.1%	0.0%
	CC	0.0%	0.0%	0.0%
YUV	TGM	-40.8%	-29.7%	-17.2%
	M	-24.7%	-18.7%	-10.9%
	A	0.2%	0.4%	0.1%
	CC	0.0%	0.0%	0.1%

TABLE VI
BD-RATE CHANGES FOR SCM5.2 WITH PALETTE VERSUS WITHOUT PALETTE (INTRA BLOCK COPY IS ENABLED)

Cat.		AI	RA	LD-B
RGB	TGM	-21.3%	-16.2%	-11.0%
	M	-6.8%	-5.3%	-3.1%
	A	-0.1%	-0.1%	-0.1%
	CC	0.0%	0.0%	0.0%
YUV	TGM	-23.1%	-17.6%	-11.4%
	M	-9.9%	-8.3%	-5.3%
	A	0.2%	0.3%	0.0%
	CC	0.0%	0.0%	0.0%

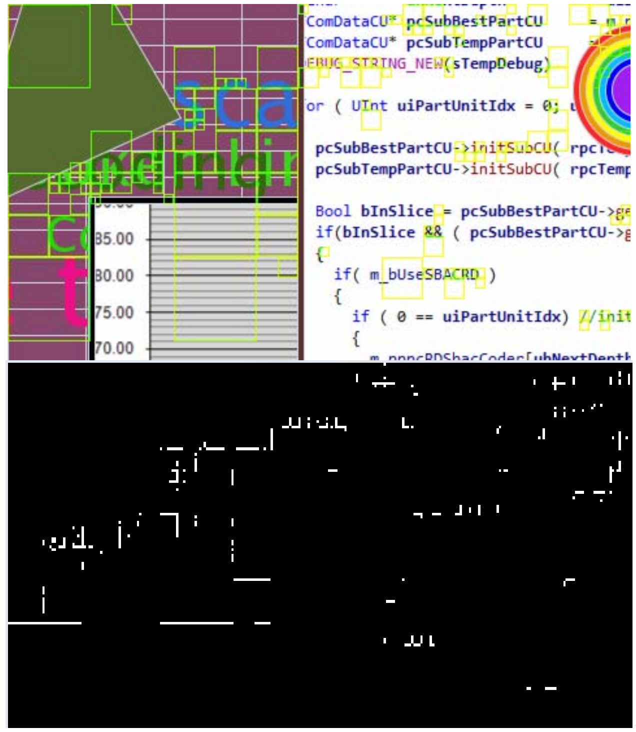


Fig. 8. Top) Part of the first frame of sc_flyingGraphics_rgb at QP 22 with the palette coded areas highlighted in green/yellow boxes. Bottom) White pixels show the locations modified by the deblocking filter in palette mode.

B. Coding Efficiency Results

Palette and intra block copy are two of the major new coding tools in HEVC SCC. In this section, palette mode's performances with intra block copy disabled and enabled are presented in Tables V and VI, respectively. When intra block copy mode is enabled, the search range is restricted to 4 CTUs. Only the BD-rates of the first color component are presented due to space limit.

Palette mode is effective in removing the redundancies in a block of pixels with strong dominate colors. Therefore, from Tables V and VI, it is observed that the palette mode can substantially reduce the BD-rates for sequences from classes TGM and M. On the other hand, palette mode does not improve the coding efficiency for animation and camera captured sequences as these test sequences (i.e., class A and CC) resemble camera captured content (non-screen content) and do not have localized dominate colors. Based on these results, we can see palatte mode is a good new coding tool for screen content based applications such as massive open online course (MOOC) with content primarily from classes TGM and M.

The performance of each individual algorithm presented in Section II and III can be found in their corresponding JCT-VC proposals. An important issue which was not discussed in the previous sections is the interaction between palette mode and deblocking filter. In HEVC, deblocking filter was used to reduce to the blocking artifacts primarily due to transform coding. However, transform is not applied to palette mode. Therefore, filtering palette mode can introduce undesirable blur to the preferable shape edges in palette mode coded blocks. As an example, Fig. 8 shows the areas distorted by deblocking filter in palette mode blocks from sc_flyingGraphics_rgb test sequence. More examples can be found in [65]. To avoid such blur, in HEVC SCC, deblocking filter is normatively disabled for palette mode.

V. CONCLUSION

This paper presents an overview of the architecture as well as the key technologies in palette mode in the HEVC SCC. Palette mode is a new coding tool targeting on compressing screen content with limited number of unique colors within a CU block. It achieves a substantial BD-rate reduction with marginal memory bandwidth increment and limited computational complexity increment. It does not require access to the neighboring CUs' reconstructed pixels so that the memory access complexity is minimized. Its decoding operations are mainly table look up and inverse quantization without inverse transform so that the computational complexity is lower than the normal transform based intra or inter mode.

Simulation results under SCC common test conditions are presented. BD rate reductions of 21.3%, 16.2%, 11.0% are achieved on average for TGM RGB sequences under AI, RA, LD-B configurations, respectively. The palette mode, together with other coding tools in HEVC SCC, constitutes a powerful codec for many emerging application such as online education and virtual desktop.

REFERENCES

[1] 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.

[2] ISO/IEC JTC1/SC29/WG11, *Joint Call for Proposals for Coding of Screen Content*, document MPEG2014/N14157, ITU-T Q6/16 Visual Coding and ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Audio, San Jose, CA, USA, Jan. 2014.

[3] 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.

[4] 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.

[5] 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.

[6] X. Guo, Y. Lu, and S. Li, *RCE4 Test1: Major-Color-Based Screen Content Coding*, document JCTVC-P0108, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 16th Meeting, San Jose, CA, USA, Jan. 2014.

[7] L. Guo, W. Pu, M. Karczewicz, J. Sole, R. Joshi, and F. Zou, *RCE4: Results of Test 2 on Palette Mode for Screen Content Coding*, document JCTVC-P0198, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 16th Meeting, San Jose, CA, USA, pp. 9–17 Jan. 2014.

[8] Y.-C. Sun *et al.*, *AHG10: A Triplet Palette Mode Combining JCTVC-P0108 and JCTVC-0198*, document JCTVC-Q0083, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[9] 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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[10] T.-D. Chuang, C.-Y. Chen, Y.-C. Sun, Y.-W. Huang, and S. Lei, *CE1-Related: Index Map Scan for 64x64 Palette Coding Block*, document JCTVC-T0058, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[11] R. Joshi *et al.*, *High Efficient Video Coding (HEVC) Screen Content Coding: Draft 5*, document JCTVC-V1005, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 22nd Meeting, Geneva, Switzerland, Oct. 2015.

[12] M. Karczewicz, F. Zou, V. Seregin, W. Pu, and R. Joshi, *CE1 Related: Improved Palette Encoder*, document JCTVC-U0096, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 21st Meeting, Warsaw, Poland, Jun. 2015.

[13] C.-H. Hung, C.-L. Lin, Y.-Y. Chang, C.-C. Lin, and J.-S. Tu, *CE1 Related: Improved Palette Table Generation*, document JCTVC-T0083 Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[14] W. Pu, X. Guo, P. Onno, P. Lai, and J. Xu, *AHG10: Suggested Software for Palette Coding Based on RExt6.0*, document JCTVC-Q0094, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 17th Meeting, Valencia, Spain, Mar. Apr. 2014.

[15] HEVC-SCC Reference Software Version 5.2 (SCM-5.2). [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.6+SCM-5.2/

[16] C. Gisquet, G. Laroche, and P. Onno, *AhG10: Palette Predictor Stuffing*, document JCTVC-Q0063, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[17] Y.-C. Sun *et al.*, *SCCE3 Test A.2: Palette Coding and Palette Predictor Updates*, document JCTVC-R0119, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[18] C. Gisquet, G. Laroche, and P. Onno, *SCCE3: Test A.3—Palette Stuffing*, document JCTVC-R0082, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[19] R. Joshi and X. Xiu, *CE5: Summary Report of Core Experiment 5 on Investigation of Maximum Palette Size and Maximum Palette Predictor Size*, document JCTVC-S0025, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[20] Y.-C. Sun, T.-D. Chuang, P. Lai, Y.-W. Chen, S. Liu, and Y.-W. S. Huang Lei, "Palette mode—A new coding tool in screen content coding extensions of HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Quebec City, QC, Canada, Sep. 2015, pp. 27–30.

[21] C. Gisquet, T. Poirier, G. Laroche, and P. Onno, *Non-CE1: On Palette Prediction for Slices*, document JCTVC-T0048, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Geneva, Feb. 2015.

[22] B. Li and J. Xu, *On WPP With Palette Mode and Intra BC Mode*, document JCTVC-S0088, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[23] K. Misra, S. H. Kim, and A. Segall, *Using the Wavefront Store-and-Sync Design for Palette Table Prediction Variables*, document JCTVC-S0141, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[24] H. Yu, R. Cohen, K. Rapaka, and J. Xu, *Common Test Condition for Screen Content Coding*, JCTVC-U1015, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 21st Meeting, Warsaw, Poland, Jun. 2015.

[25] M. Karczewicz, W. Pu, V. Seregin, R. Joshi, and J. Sole, *CE6: Test D.1 Run-Length Coding for Reuse Flags*, document JCTVC-S0154, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[26] 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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[27] Y.-C. Sun, T.-D. Chuang, Y.-W. Chen, X. Zhang, Y.-W. Huang, and S. Lei, *SCCE3 Test B.4: Adaptive Color Index Map Scan*, document JCTVC-R0057, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[28] W. Pu, R. Joshi, M. Karczewicz, J. Sole, and V. Seregin, *SCCE3: Test B. 13—Flag to Indicate Escape and Index Coding*, document JCTVC-R0066, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[29] Wikipedia. *Truncated Binary Code*. [Online]. Available: https://en.wikipedia.org/wiki/Truncated_binary_encoding

[30] W. Pu, V. Seregin, R. Joshi, M. Karczewicz, and J. Sole, *SCCE3: Test B. 12—Binarization of Escape Sample and Palette Index*, document JCTVC-R0065, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[31] B. Li and J. Xu, *On the Palette Escape Pixel Coding*, document JCTVC-U0052, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 21st Meeting, Warsaw, Poland, Jun. 2015.

[32] Y.-C. Sun, T.-D. Chuang, Y.-W. Chen, Y.-W. Huang, and S. Lei, *Non-RCE4: Removal of Syntax Redundancy in RCE4 Test2*, document JCTVC-P0095, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 16th Meeting, San Jose, CA, USA, Jan. 2014.

[33] W. Pu, F. Zou, V. Seregin, R. Joshi, and M. Karczewicz, *Non-CE6: Palette Parsing Dependency and Palate Encoder Improvement*, document JCTVC-S0157, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[34] 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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[35] 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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[36] S.-T. Hsiang, T.-D. Chuang, and S. Lei, *CE6: Results of Test A.4 on Palette Run Coding*, document JCTVC-S0163, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[38] R. Joshi, W. Pu, M. Karczewicz, F. Zou, V. Seregin, and J. Sole, *CE6 Subtest A.6: Binarization for Run Coding in Palette Mode*, document JCTVC-S0039, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[39] R. Joshi, M. Karczewicz, and W. Pu, *Non-SCCE3: Contexts for Coding Index Runs*, document JCTVC-R0174, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 18th Meeting, Sapporo, Japan, Jun./Jul. 2014.

[40] R. Joshi, M. Karczewicz, W. Pu, V. Seregin, J. Sole, *CE6 Subtest A.5: Contexts for Run Coding in Palette Mode*, document JCTVC-S0038, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[41] W. Pu, M. Karczewicz, F. Zou, V. Seregin, and J. Sole, *Non-CE6: Improvement on Palette Run Coding*, document JCTVC-S0111, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[42] M. Karczewicz, R. Joshi, W. Pu, V. Seregin, and F. Zou, *Non-CE1: Modification of Palette Run Coding*, document JCTVC-T0133, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[43] W. Pu, F. Zou, R. Joshi, M. Karczewicz, and J. Sole, *Improvements to Palette Based Coding*, document JCTVC-Q0225, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 17th Meeting, Valencia, Spain, Mar./Apr. 2014.

[44] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based advanced binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 620–636, Jul. 2003.

[45] J. Sole *et al.*, "Transform coefficient coding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1765–1777, Dec. 2012.

[46] 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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[47] X. Xiu, Y. He, and Y. Ye, *Removal of Parsing Dependency in Palette-Based Coding*, document JCTVC-S0181, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[48] M. Xu, Z. Ma, W. Wang, and H. Yu, *Non-CE6: Simplification on Escape Coding of Palette Mode in HEVC SCC*, document JCTVC-S0181, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[49] T.-D. Chuang, C.-Y. Chen, Y.-W. Huang, and S. Lei, *CE1-Related: Colour-Plane-Based Escape Pixel Coding*, document JCTVC-U0063, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 21st Meeting, Warsaw, Poland, Feb. 2015.

[50] W. Pu, R. Joshi, T. Hsieh, M. Karczewicz, F. Zou, and V. Seregin, *Syntax Cleanup for Palette Mode*, document JCTVC-U0087, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 21st Meeting, Warsaw, Poland, Feb. 2015.

[51] W. Pu, R. Joshi, V. Seregin, M. Karczewicz, and F. Zou, *Non 4:4:4 Palette Mode: AHG Way*, document JCTVC-T0062, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[52] R. Joshi, W. Pu, V. Seregin, M. Karczewicz, and F. Zou, *Non-CE1: Extension of Palette Mode to Non-4:4:4 Formats*, document JCTVC-T0109, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[53] J. Ye, S. Liu, and S. Lei, *CE1-Related: Palette Coding for Non-444 Format Content*, document JCTVC-T0072, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[54] F. Zou, W. Pu, M. Karczewicz, R. Joshi, V. Seregin, and J. Sole, *CE6: Test C.3 Copy Previous Row Mode for Palette Coding*, document JCTVC-S0173, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[55] 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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 16th Meeting, San Jose, CA, USA, Jan. 2014.

[56] 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 (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 20th Meeting, Geneva, Switzerland, Feb. 2015.

[57] Y.-C. Sun *et al.*, *CE1: Test A.1: Extended Copy Above Mode to the First Line With Index Adjustment Bits*, document JCTVC-U0061, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 21st Meeting, Warsaw, Poland, Feb. 2015.

[58] Y.-C. Sun *et al.*, *CE1: Test A.2: Extended Copy Above Mode to the First Line With Grouping the Palette_Run_Type_Flag in Front*, document JCTVC-U0062, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 21st Meeting, Warsaw, Poland, Feb. 2015.

[59] J. Ye and S. Liu, *Non-CE6: Copy Previous Mode*, document JCTVC-S0120, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[60] C. Gisquet, G. Laroche, and P. Onno, *Non-RCE4: Transition Copy Mode for Palette Mode*, document JCTVC-P0115, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 16th Meeting, San Jose, CA, USA, Jan. 2014.

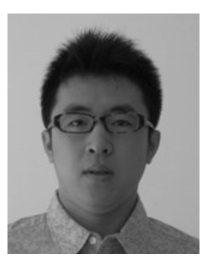
[61] Y.-C. Sun, T.-D. Chuang, Y.-W. Chen, Y.-W. Huang, and S. Lei, *CE6 Test C.2: Transition Copy Mode*, document JCTVC-S0078, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[62] W. Wang, Z. Ma, M. Xu, and H. Yu, *Non-CE6: 2-D Index Map Coding of Palette Mode in HEVC SCC*, document JCTVC-S0151, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[63] Y.-C. Sun, T.-D. Chuang, Y.-W. Chen, Y.-W. Huang, and S. Lei, *Non-CE6: A Combination of CE6 Test C.2—Transition-Copy Mode and CE6 Test C.3 (Configuration 1)—Copy-From-Previous-Row Mode*, document JCTVC-S0188, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.

[64] G. Bjøntegaard, *Improvements of the BD-PSNR Model*, document ITU-T SG16 Q.6, VCEG-AI11, Berlin, Germany, Jul. 2008.

[65] J. Sole, W. Pu, C. Pang, R. Joshi, V. Seregin, and M. Karczewicz, *AhG13: Palette and Deblocking*, document JCTVC-S0096, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 19th Meeting, Strasbourg, France, Oct. 2014.



Wei Pu (M'08–SM'15) received the B.S. and Ph.D. degrees in computer engineering from University of Science and Technology of China (USTC), Hefei, China, in 2004, and 2009, respectively.

Currently, he is a software engineer on machine learning in Uber Advanced Technologies Center, Pittsburgh, PA, USA. From 2010 until 2012, he was a research scientist in Department of Computer Science and Engineering, State University of New York at Buffalo. From 2012 until 2015, he was with Qualcomm Inc. He has over 100 publications, over 50 granted/pending patents on image and video processing, compression, and streaming. His current research interests are robotics and machine learning.

Marta Karczewicz received the M.Sc. and Ph.D. (Hons.) degrees from the Tampere University of Technology, Tampere, Finland, in 1994 and 1997, respectively.

From 1996 to 2006, she was with Nokia Research, Tampere. She is currently the Vice Present of the Multimedia Research and Development and Standards Group, Qualcomm, San Diego, CA, USA.

Rajan Joshi (M'95) received the B.Tech. degree in electrical engineering and the M.Tech. degree in communications engineering from the Indian Institute of Technology Bombay, Mumbai, India, in 1988 and 1990, respectively, and the Ph.D. degree in electrical engineering from Washington State University, Pullman, WA, USA, in 1996.

In 1995, he was with the Xerox Palo Alto Research Center, Palo Alto, CA, USA. From 1996 to 2006, he was a Senior Research Scientist with Eastman Kodak Company, Rochester, NY, USA. From 2006 to 2008, he was a Senior Technical Staff Member with Thomson Corporate Research, Burbank, CA, USA. He is currently a Senior Staff Engineer with Qualcomm, San Diego, CA, USA. His current research interests include video and image coding, video processing, and information theory.

Vadim Seregin received the M.S. (honors) degree in physics and the Ph.D. degree in mathematical modeling, numerical methods, and software systems from Moscow State University, Moscow, Russia, in 2002 and 2005, respectively.

Currently, he is a Senior Staff Engineer with Qualcomm, San Diego, CA, USA. From 2006 to 2011, he was a Senior Engineer with the Multimedia Platform Laboratory, Digital Media and Communications Research and Development Center, Samsung Electronics, Suwon, Korea. His research interests include image and video coding, video processing, and transmission. He has been an active contributor to HEVC and HEVC extensions standardization activities.

Feng Zou (S'08–M'14) received the B.S. degree in electrical engineering from Harbin Institute of Technology, China, in 2008, and the Ph.D. degree in electronic and computer engineering in the Hong Kong University of Science and Technology (HKUST), Hong Kong SAR in 2013. While at HKUST, he was supervised by Prof. O. C. Au.

From June 2012 to June 2013, he interned in Mitsubishi Electric Research Labs (MERL) Cambridge, MA, USA. After that, he worked at Qualcomm as a Senior Engineer in San Diego, and his current research interests include intra prediction, transform, quantization designs, 3D video coding and screen content coding. He is actively contributing proposals in the standardization of HEVC under the ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), JCT3V and range extension of HEVC and holds several video coding related patents.

Joel Sole (M'01–SM'14) received the M.Sc. degree in telecommunications from the Technical University of Catalonia (UPC), Barcelona, Spain, and Telecom ParisTech, Paris, France, and the Ph.D. degree from UPC, in 2006.

He was with Thomson Corporate Research, Princeton, NJ, USA, from 2006 to 2010, initially as a Post-Doctoral Fellow, then as Staff Member, and later as Senior Scientist. Since 2010, he has been with Qualcomm Inc., San Diego, CA, USA, as a Senior Staff Engineer and Manager, where he has led projects related to video coding. He has participated in several standardization activities in MPEG and JCT-VC, including H.265/HEVC, HEVC RExt, Screen Content Coding, and High Dynamic Range video coding.

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). His research interests include image/video processing, compression, and communication.

Tzu-Der Chuang was born in Taipei, Taiwan in 1983. He received the B.S. degree from the Department of Electrical Engineering, National Taiwan University (NTU) in 2005 and the Ph.D. degree from the Graduate Institute of Electronics Engineering (GIEE), NTU in 2010.

In 2010, he joined MediaTek, and currently he is a technical manager. His research interests include image and video coding, image and video processing, and related VLSI architectures. In 2010, he started attending international video coding standard meetings held by ISO/IEC MPEG and ITU-T VCEG. He contributed over 120 technical proposals for the standardization of the High Efficiency Video Coding (HEVC) and the Extensions of HEVC.

Dr. Chuang received the 47th DAC/ISSCC Student Design Contest in 2010, the first prize of the eight Golden Silicon Awards (founded by Macronix), and the 2012 MediaTek Innovation Award (MediaTek-internal). He is a member of Phi Tau Phi Scholastic Honor Society.

Polin Lai received the B.S. degree from National Taiwan University, in 2002, and the M.S. and Ph.D. degrees from the University of Southern California (USC), Los Angeles, CA, USA, in 2004 and 2008, respectively, all in electrical engineering. He was awarded USC Doctoral Scholarship.

From 2008 to 2010, he was a Member of Technical Staff with Technicolor Research and Innovation, Princeton, NJ, USA, working on 3D video compression and processing. From 2010 to 2012, he was a Senior Standards Engineer with Samsung Research America, Dallas, TX, USA, focusing on video coding standardization including HEVC. Since 2012, he has been with MediaTek USA Inc., San Jose, CA, USA. As a Senior Member of Technical Staff and a team lead, he continued working on standardization including scalable video and screen content coding, and now also on streaming and delivery of VR video.

Shan Liu (M'02–SM'11) received the B.E. degree in electronic engineering from Tsinghua University, Beijing, China, in 1996, the M.S. and Ph.D. degrees in electrical engineering from University of Southern California, Los Angeles, CA, USA, in 1999 and 2002, respectively.

From 2000 to 2002, she held visiting research positions at AT&T Labs, Rockwell Science Center and IBM Watson Research Center. From 2003 to 2010, she held senior technical positions at Samsung America, Sony America and Mitsubishi Electric Research Labs. She is currently a director of technology development at MediaTek (USA) Inc. She has published more than 30 peer-reviewed papers, more than 50 patents and more than 200 contributions to ISO/IEC and ITU-T standards. She was an editor for HEVC ver. 3.

Dr. Liu received the Outstanding Academic Award from University of Southern California in 1999 and R&D Gold Award from MediaTek in 2012.

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.

Jing Ye received the B.S. degree in electronic engineering from Xidian University, Xi'an, Shaanxi, China, in 2010. She received the M.S. degrees in electrical engineering from University of Southern California, Los Angeles, CA, USA, in 2012.

From 2013 she has been working on next generation video coding standard, such as HEVC and its extensions in Mediatek. Her current research interests include video processing and video compression.

Yu-Wen Huang was born in Kaohsiung, Taiwan, in 1978. He received the B.S. degree from the Department of Electrical Engineering, National Taiwan University (NTU), in 2000, and the Ph.D. degree from the Graduate Institute of Electronics Engineering (GIEE), NTU, in 2004.

In 2004, he joined MediaTek, and currently he is a Senior Department Manager. From 2004 to 2006, he was an integrated circuit designer developing video encoding algorithms, architectures, and related hardware circuits for products. Since 2007, he has been researching on future video coding technologies. Since 2010, he has been in charge of the Video Coding and Processing department of the Multimedia Technology Development division. In 2006, he started attending international video coding standard meetings held by ISO/IEC MPEG and ITU-T VCEG, and up to now he remains active in the standard meetings. From 2010 to 2015, his team contributed over 300 technical proposals for the standardization of the High Efficiency Video Coding (HEVC) and the Extensions of HEVC. His research interests include image and video coding, image and video processing, and related VLSI architectures.

Dr. Huang received the 2005 Student Outstanding Research Award (from GIEE, NTU), the Gold Medal of 2005 Long-Term Thesis Award (founded by Acer), the 2006 Excellent Technology Transfer Contribution Award (from Ministry of Science and Technology, Taiwan), the 2007–2011 MediaTek Best Paper Awards (MediaTek-internal), and the 2012 MediaTek Innovation Award (MediaTek-internal). He is an author or coauthor of over 25 international journal papers, over 40 international conference papers, over 30 granted US patents, and over 40 pending U.S. patent applications. He is a member of Phi Tau Phi Scholastic Honor Society.